1974-09

A computer code for solving medium sized non-linear programming problems by the method of feasible directions.

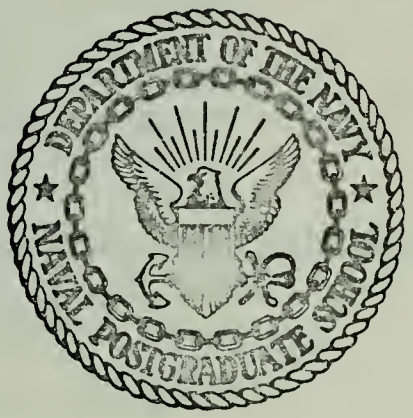
Harrison, James Douglas.

Monterey, California. Naval Postgraduate School

A COMPUTER CODE FOR SOLVING MEDIUM SIZED NON-LINEAR PROGRAMMING PROBLEMS BY THE METHOD OF FEASIBLE DIRECTIONS

James Douglas Harrison

NAVAL POSTGRADUATE SCHOOL
Monterey, California



THESIS

A COMPUTER CODE FOR SOLVING MEDIUM SIZED NON-LINEAR PROGRAMMING PROBLEMS BY THE METHOD OF FEASIBLE DIRECTIONS

by

James Douglas Harrison

September 1974

Thesis Advisor: G. T. Howard

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Date Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)

A Computer Code for Solving Medium Sized Non-Linear Programming Problems by the Method of Feasible Directions

5. TYPE OF REPORT & PERIOD COVERED

Master's Thesis; September 1974

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)

James Douglas Harrison

8. CONTRACT OR GRANT NUMBER(s)

9. PERFORMING ORGANIZATION NAME AND ADDRESS

Naval Postgraduate School
Monterey, California 93940

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS

Naval Postgraduate School
Monterey, California 93940

12. REPORT DATE

September 1974

13. NUMBER OF PAGES

70

14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)

Naval Postgraduate School
Monterey, California 93940

15. SECURITY CLASS. (of this report)

Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Non-Linear Programming
Computer-Code
Maximization
Constrained Optimization
Feasible Directions

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A computer code, FEASBL, is developed to maximize a non-linear objective function over a convex feasible region. The feasible region is defined by a set of non-linear and linear constraints on the variables. FEASBL can solve problems involving up to fifty variables with a feasible region formed by up to fifty non-linear

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601 |
(Page 1)

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Date Entered)

Block #20 continued

constraints, and fifty linear constraints. FEASBL uses a feasible direction method as its solution algorithm.

A Computer Code for Solving Medium Sized Non-Linear Programming Problems by the Method of Feasible Directions

by

James Douglas Harrison
Lieutenant Commander, United States Navy
B.Chem., University of Minnesota, 1963

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
September 1974

ABSTRACT

A computer code, FEASBL, is developed to maximize a non-linear objective function over a convex feasible region. The feasible region is defined by a set of non-linear and linear constraints on the variables. FEASBL can solve problems involving up to fifty variables with a feasible region formed by up to fifty non-linear constraints, and fifty linear constraints. FEASBL uses a feasible direction method as its solution algorithm.

TABLE OF CONTENTS

I. INTRODUCTION

This thesis presents a computer code for solving medium sized non-linear programming problems, and describes its use at the Naval Postgraduate School computer center.

The general non-linear programming problem is the optimization of an objective function over a constrained feasible region. By the technique of separable programming, many non-linear problems can be converted to a linear problem and solved using established linear programming methods. Many problems, however, cannot be solved by this method. Several techniques have been developed for solving non-linear problems. Most methods involve following the gradient of the objective function to generate a sequence of points that converges to the optimal point. The method used in this code is Zoutendijk's method of feasible directions [1]. The method of feasible directions can be considered a large-step gradient method as opposed to the small-step gradient and search methods. Feasible direction methods can be characterized by:

1. Starting at an initial feasible point.

2. Establishing a feasible direction that will improve the objective function.

3. Proceeding to a new feasible point that has a better value of the objective function than the previous point.

4. Converging to the optimal point, or determining that the problem is unbounded.

Many procedures can be considered feasible directions methods, for example, the simplex method for solving linear programming problems.

The method used in this code requires that the user know the objective function, constraint functions, all their first partial derivatives, and an initial feasible point. If the user does not know an initial feasible point, this code can be used to solve a preliminary programming problem which will provide a feasible point. This problem will be described later. This code will solve maximization problems of the form:

$$\text{Max:} \quad z = F(\underline{x})$$

$$\text{Subj to:} \quad G_i(\underline{x}) \leq b_i \ , \quad i \in I_C$$

$$a_i^T \underline{x} \leq b_i \ , \quad i \in I_L$$

$$d_j \leq x_j \leq c_j \ , \quad j \in J \ .$$

The functions $F(\underline{x})$, and $G_i(\underline{x})$ are non-linear functions which are supplied by the user as Fortran FUNCTION routines. The sets I_C and I_L are the sets of subscripts for the non-linear, and linear constraints respectively. If the objective function is concave, and the non-linear constraints are convex functions, then the problem will be a convex programming problem, and the method will converge to the global maximum. If the problem is not convex, then the method may converge to a local maximum. The code is designed to accommodate functions with up to fifty variables. The constraint set

can include up to fifty non-linear constraints and fifty linear constraints. A complete description of how to use this code is provided in Section IV.

II. MATHEMATICAL BACKGROUND

The theoretical foundation for the method of feasible directions was developed by Zoutendijk in 1960 [1]. A brief discussion of the mathematical background as it applies to the method used in this computer code is presented.

A. DEFINITIONS

A region R is a convex region if for $\underline{x}_1$ and $\underline{x}_2$, two points in R, then the point $\underline{x}_3 = a\underline{x}_1 + (1 - a)\underline{x}_2$ is in R for all a such that $0 \leq a \leq 1$.

Let R be the feasible region, and if $\underline{x}$ is a member of R then a direction $\underline{s}$ is a feasible direction if and only if there exists a $\lambda' > 0$ such that $\underline{x} + \lambda s$ is in R for all $\lambda \varepsilon (0,\lambda')$.

A usable feasible direction is a feasible direction $\underline{s}$ such that $\nabla_x F(\underline{x})^T s > 0$. In other words, a usable feasible direction at $\underline{x}$ is a direction in which the value of the objective function increases and along which it is possible to move while remaining in R.

Let $F(\underline{x})$ be a concave function, and let the feasible region R be convex. Then if there exists no usable feasible direction at $\underline{x}$ in R, then $\underline{x}$ must be a maximum of $F(\underline{x})$ over the region R. To show this assume that $\underline{x}$ is not a maximum and there is no usable feasible direction at $\underline{x}$. There exists a point $\underline{y}$ in R such that $F(\underline{y}) > F(\underline{x})$ and since R is convex all the points along the line between $\underline{y}$ and $\underline{x}$ must be in R.

Further, since F( ) is concave, $\nabla_x F(\underline{x})^T(\underline{y} - \underline{x}) > 0$, thus the direction $\underline{y} - \underline{x}$ is a usable feasible direction. This contradiction implies that $\underline{x}$ must be a maximum of $F(\underline{x})$.

B. THE METHOD

The method of feasible directions begins at an initial feasible point x^o, determines the "best" usable feasible direction, moves along that direction to the point $\underline{x}^k$ that maximizes $F(\underline{x})$ along the direction without leaving the feasible region, repeats this procedure until a point $\underline{x}^*$ is found which has no usable direction, or determines that the problem is unbounded. The method can thus be viewed as a four step procedure.

1. Determine a usable feasible direction $\underline{s}^k$ at the point $\underline{x}^k$.

2. If no usable feasible direction exists stop, $\underline{x}^k$ is the maximum of $F(\underline{x})$ over R.

3. Determine a step length λ_k such that $\underline{x}^{k+1} = \underline{x}^k + \lambda_k \underline{s}^k$ is in R and $\underline{x}^{k+1}$ maximizes $F(\underline{x})$ along $\underline{s}^k$.

4. If λ_k goes to infinity, then the problem is unbounded, otherwise go back to step 1 and repeat.

The direction finding problem will be discussed first, and then the step length problem.

1. The Direction Finding Problem

The direction finding problem is to find the usable feasible direction that will result in the greatest apparent increase in the objective function. This can be formulated as the following linear programming problem.

$$\text{Max:} \quad \sigma$$

$$\text{Subj to:} \quad \sigma - \nabla_x F(\underline{x})^T \underline{s} \leq 0$$

$$\sigma + \nabla_x G_i(\underline{x})^T \underline{s} \leq 0, \; i \in I_c(\underline{x})$$

$$\underline{a}_i^T \underline{s} \leq 0, \; i \in I_1(\underline{x})$$

$$s_j \geq 0, \; j \in J^-(\underline{x})$$

$$s_j \leq 0, \; j \in J^+(\underline{x})$$

$$|s_j| \leq 1, \; j \in J(\underline{x}),$$

where $I_c(\underline{x})$ and $I_1(\underline{x})$ are the sets of binding non-linear constraints and binding linear constraints, and $J^-(\underline{x})$ and $J^+(\underline{x})$ are the sets of j such that x_j is at its lower and upper bound respectively.

If there exists at least one point that is strictly interior to R with respect to the non-linear constraints and if the optimal value of σ is 0 in the direction finding problem, then there is no usable feasible direction and we are at the optimum point for the non-linear problem. The last constraint in the direction finding problem is a normalization constraint. In order to insure that the procedure will converge the components of $\underline{s}^k$ must be bounded, and several normalizations have been studied by Zoutendijk. These normalizations are:

$$\text{N1:} \quad \underline{s}^T \underline{s} \leq 1$$

$$\text{N2:} \quad |s_j| \leq 1$$

N3: $s_j \leq 1$ if $\nabla_x F(\underline{x})_j > 0$

$s_j \geq -1$ if $\nabla_x F(\underline{x})_j < 0$

N4: $\sigma \leq 1$

N5: $\nabla_x G_i(\underline{x})^T \underline{s} + \sigma \leq b_i - G_i(\underline{x})$; $i \varepsilon I_c(\underline{x})$

$\underline{a}_i^T \underline{s} \leq b_i - \underline{a}_i^T \underline{x}$; $i \varepsilon I_L(\underline{x})$

$d_j - x_j \leq s_j \leq c_j - x_j$.

The normalization N1 will generally lead to fewer steps then N2, N3, or N4, but the amount of computation for each step is considerably greater. Also the simple constraints $s_j \leq 0$, and $s_j \geq 0$ increases the size of the direction finding problem if N1 is used, but not for N2, N3, and N4. As one goes from N2, to N4 the number of steps needed increases, but the amount of computation for each step decreases. N5 leads to a larger problem, and also does not converge as fast as the other normalizations. Normalization N2 was chosen for this code as the best compromise between the number of steps and the amount of work for each step.

2. The Modified Direction Finding Problem

To insure that the procedure actually converges to the maximum, some precautions must be taken to prevent the phenomenon of "jamming" in which the procedure in effect gets "stuck in a corner" and never finds the real maximum. This is accomplished by considering nearby constraints in addition to the binding constraints when solving the direction finding problem.

Define the sets:

$$I_C(\underline{x},\varepsilon) = \{i\varepsilon I_C : b_i - \varepsilon \leq G_i(\underline{x}) \leq b_i\}$$

$$I_L(\underline{x},\varepsilon) = \{i\varepsilon I_L : b_i - \varepsilon \leq \underline{a}_i^T \underline{x} \leq b_i\}$$

$$J^-(\underline{x},\varepsilon) = \{j\varepsilon J : d_j \leq x_j \leq d_j + \varepsilon\}$$

$$J^+(\underline{x},\varepsilon) = \{j\varepsilon J : c_j - \varepsilon \leq x_j \leq c_j\}.$$

The direction finding problem is then modified to include the constraints in the sets $I_C(\underline{x},\varepsilon)$, $I_L(\underline{x},\varepsilon)$, $J^-(\underline{x},\varepsilon)$, and $J^+(\underline{x},\varepsilon)$. Therefore the direction finding problem becomes:

$$\begin{aligned}
&\text{Max:} && \sigma \\
&\text{Subj to:} && \sigma - \nabla_x F(\underline{x})^T \underline{s} \leq 0 \\
& && + \nabla_x G_i(\underline{x})^T \leq 0 \quad , \; i\varepsilon I_C(\underline{x},\varepsilon) \\
& && \underline{a}_i^T \underline{s} \leq 0 \quad , \; i\varepsilon I_L(\underline{x},\varepsilon) \\
& && s_j \geq 0 \quad , \; j\varepsilon J^-(\underline{x},\varepsilon) \\
& && s_j \leq 0 \quad , \; j\varepsilon J^+(\underline{x},\varepsilon) \\
& && |s_j| \leq 1 \quad , \; j\varepsilon J(\underline{x}) \; .
\end{aligned}$$

Then if the solution to the direction finding problem is such that $\sigma < \varepsilon$, the problem is solved again with $\varepsilon/2$. Thus we stay away from the "corners" until we get near the optimal point.

3. The Step Length Problem

Once the direction $\underline{s}^k$ has been found, the problem is to determine how far to move in that direction. If λ_F is defined to be the largest step length λ such that $\underline{x}^k + \lambda \underline{s}^k$

is in R, then the step length will be λ_k such that $F(\underline{x}^k + \lambda_k \underline{s}^k)$ is the maximum of $F(\underline{x}^k + \lambda \underline{s}^k) : \lambda \leq \lambda_F$. If $\lambda_k = \infty$, then the problem is unbounded, and we can conclude that the maximum of $F(\underline{x})$ is infinite. The maximum feasible step λ_F is found by a simple search along the direction $\underline{s}^k$ from the point $\underline{x}^k$. Then the optimal step length λ_k is determined by finding the λ such that $\nabla_x F(\underline{x}^k + \underline{s}^k)\, \underline{s}^k = 0$. Since $F(\underline{x})$ is concave, if $\nabla_x F(\underline{x}^k + \lambda_F \underline{s}^k)^T\, \underline{s}^k > 0$, then λ_F is used for the step length, otherwise λ_k can be found by *regula falsi* [2].

4. Stopping Conditions

The procedure will terminate whenever the solution to the direction finding problem yields $\sigma = 0$, indicating that there are no usable feasible directions. If the solution to the step length problem is $\lambda_k = \infty$, then the problem is unbounded and the procedure is terminated indicating an unbounded solution. In addition, if the step length or the percent improvement in the objective function becomes smaller than user provided limits, the problem is terminated, and the current solution is specified.

C. THE INITIAL FEASIBLE POINT

If an initial feasible point is not known, pick an $\underline{x}'$ such that $d_j \leq x'_j \leq c_j$, and solve the problem:

$$\text{Max:} \quad -z_1$$

$$\text{Subj to:} \quad \underline{a}_i^T \underline{x} - p_i z_1 \leq b_i \; ; \; i \in I_L$$

$$d_j \leq x_j \leq c_j$$

$$z_1 \geq 0$$

where

$$p_i = \begin{cases} 0 \ ; \text{ if } a_i^T \underline{x}' \leq b_i \\ a_i^T \underline{x}' - b_i \ ; \text{ if } a_i^T \underline{x}' > b_i. \end{cases}$$

An initial feasible point for this linear programming problem is $\underline{x}'$, $z_1 = 1$. If the feasible region is non-empty, then the optimal solution of this problem will be $z_1 = 0$, $\underline{x} = \underline{x}''$. Now $\underline{x}''$ will be feasible with respect to the linear constraints, if $\underline{x}''$ is also feasible with respect to the non-linear constraints, then use $\underline{x}''$ for the initial feasible point. However if $\underline{x}''$ for the initial feasible point. However if $\underline{x}''$ is not feasible with respect to the non-linear constraints, solve the problem:

Max: $-z_2$

Subj to: $G_i(\underline{x}) - p_i z_2 \leq b_i \ ; \ i \in I_C$

$a_i^T \underline{x} \leq b_i \ ; \ i \in I_L$

$d_j \leq x_j \leq c_j$

$z_2 \geq 0$

where:

$$p_i = \left\{ \begin{array}{l} 0 \quad ; \text{ if } G_i(\underline{x}'') \leq b_i \\ \\ G_i(\underline{x}'') - b_i \ ; \text{ if } G_i(\underline{x}'') > b_i. \end{array} \right\}$$

This non-linear programming problem can be solved by the method of feasible directions using $\underline{x}''$, $z_2 = 1$ as the initial feasible point. If the feasible region of the original problem is non-empty, then the optimal solution to this problem will be $z_2 = 0$, and $\underline{x} = \underline{x}^0$. Now the original problem can be solved using $\underline{x}^0$ as the initial feasible point.

III. THE COMPUTER PROGRAM FEASBL

This section will describe the computer programming aspects of the code, a complete set of user instructions is provided in section IV. Several programming languages were considered for the program including IBM 360/FORTRAN IV, SIMSCRIPT II.5, and PL/1. FORTRAN IV was chosen because its virtually universal availability and widespread familiarity will allow easy installation at other computer centers, and facilitate user modifications in the future.

FEASBL is structured as a straight line program, consisting of several sections; input, direction finding problem building and solving, step length determination, and output. In deciding the size of the problem that the program would be able to handle, the size of the direction finding linear programming problem was the controlling factor. To insure that round-off errors would not degrade the solution, the inverse of the basis matrix is maintained in double precision. FEASBL is capable of handling fifty variables, with fifty non-linear and fifty linear constraints thus requiring a basis matrix that is 152 by 152, which requires over 184K of core. Desiring to keep the total size below 300K if possible, substantial effort was made to minimize the storage requirements throughout the program both by not storing any value that could be computed when needed, and using the same storage space for multiple purposes. The code now requires

about 290K of core space in addition to that required for user provided functions.

The information required by the code includes the problem description, the initial starting point, and some program control variables. The problem description is provided by specifying the objective function, non-linear and linear constraints, upper and lower bounds on the variables, and the requirements vectors. In addition, the first partial derivatives of the objective function and the non-linear constraint functions are required. A prototype code was developed which did not require derivatives, but which severely limited the form of the functions that could be used. It was felt, however, that the greatly increased power and flexibility gained by not restricting the form of the functions more than offset the requirement to provide the derivatives. These functions are provided in the form of Fortran FUNCTION routines, thus FEASBL can handle any function the user can express as a Fortran routine. The remaining data is input in the Fortran NAMELIST format. This method was chosen because the unstructured format minimizes keypunch errors.

Each iteration begins by building the direction finding problem. The sets $I_C(\underline{x},\varepsilon)$, $I_L(\underline{x},\varepsilon)$, $J^-(\underline{x},\varepsilon)$ and $J^+(\underline{x},\varepsilon)$ are determined, and then the direction finding linear programming problem is built. Rather then store the entire A matrix, only those elements which depend on the gradients of the objective function and non-linear constraints in $I_C(\underline{x},\varepsilon)$ referred to as the core are stored. The linear programming problem is

solved using the revised simplex algorithm [3], and the basis inverse is maintained through the product form. At each step, every non-basic variable is priced, and that variable with the most negative $z_j - c_j$ is selected for entry. In the event of ties the left-most variable is picked. The column vector $\underline{a}_{j,k}$ for the entering variable is then generated, and used to compute $\underline{y}_k = B^{-1} \cdot \underline{a}_k$. The departing variable is j such that $x_{b_j}/y_{j,k}$ is minimum for all $y_{j,k} > 0$. If there are no negative $z_j - c_j$'s, then the optimal solution has been found for the direction finding problem. After an optimal solution has been found, the basis matrix is created and inverted directly using the IBM subroutine DMINV, and the solution is checked for optimality. If the solution is not optimal, then the linear program is continued until an optimal solution is reached. When the optimal solution is achieved, the direction is available, and the program proceeds to find the best step length. This is accomplished by first finding λ_F, the largest λ such that $\underline{x}^k + \lambda \underline{s}^k$ is feasible, and then determining $\lambda_k \leq \lambda_F$ that maximizes $F(x^K + \lambda_k \underline{s}^k)$. The quantity λ_F is found by starting with $\lambda = 1$ and increasing λ by a factor of 2 until a λ_{NF} is found such that $\underline{x}^k + \lambda_{NF}\, \underline{s}^k$ is not feasible. The interval between λ_{NF} and the largest known feasible step λ_F' is reduced by checking $\lambda = \frac{1}{2}(\lambda_{NF} + \lambda_F')$ for feasibility until the interval is less then the minimum step length (STPMIN). However, if a feasible step length λ is found such that $F(\underline{x}^k + \lambda \underline{s}^k) < F(\underline{x}^k)$, then it is assumed that the maximum point along the direction

lies between these two points. Once λ_F has been determined λ_k is computed by the method of *regula falsi*. Since $F(\underline{x})$ is concave and is increasing at $\underline{x}^k$ along the direction $\underline{s}^k$ the directional derivative $(\nabla_x F(\underline{x})^T \underline{s})$ is positive at $\underline{x}^k$ and will be zero at the maximum along $\underline{s}^k$. If the directional derivative is negative at $\underline{x}^k + \lambda_F \underline{s}^k$, then the maximum lies between the two points. A new λ is then selected such that $\lambda = (b\lambda_1 - a\lambda_2) / (b - a)$ where: $a = \nabla_x F(\underline{x}^k + \lambda_1 \underline{s}^k)$; $b = \nabla_x F(\underline{x}^k + \lambda_2 \underline{s}^k)$; and λ_1 and λ_2 bracket the optimal stop length. If the directional derivative is zero at the new point the maximum has been found, otherwise we have a new interval with positive and negative directional derivatives at the endpoints, and another λ is determined. This continues until either the maximum is found, or the interval becomes less then the minimum specified. If the directional derivative is positive at $\underline{x}^k + \lambda_F \underline{s}^k$ and $F(\underline{x}^k + \lambda_F \underline{s}^k) > F(\underline{x}^k)$, then the maximum along $\underline{s}^k$ is outside the feasible region, and λ_k is set equal to λ_F. However, if $F(\underline{x}^k + \lambda_F \underline{s}^k) \leq F(\underline{x}^k)$, then the function is not concave, so a step λ_U between 0 and λ_F is found such that the directional derivative is negative, and it is assumed that $F(\underline{x})$ is concave between $\underline{x}^k$ and $\underline{x}^k + \lambda_U \underline{s}^k$, and the maximum is sought as before.

Once a usable feasible direction, and step length have been determined set $\underline{x}^{k+1} = \underline{x}^k + \lambda_k \underline{s}^k$ and repeat the process starting at $\underline{x}^{k+1}$. This process is continued until either no usable feasible direction can be found or the problem becomes unbounded. If the step length or the percent

improvement in the objective function become less then user supplied limits the program will terminate. The program will also stop after a user specified number of iterations.

IV. USING FEASBL

This section will describe how to use FEASBL, including required input, possible errors, and the normal output. An example is given illustrating different features of the code.

A. INPUT

The objective function, non-linear constraint functions, and their first partial derivatives are input as Fortran FUNCTION routines. The linear constraint matrix, requirements vectors, upper and lower bounds on x_j, the initial feasible point, and any other input is read in as data at execution.

1. The FUNCTION Routines

Four Fortran FUNCTION routines are required, these routines provide the objective function, its first partial derivatives, the non-linear constraints, and their first partial derivatives. Even if the problem has no non-linear constraints, all four FUNCTION routines must be provided. The four routines are:

FUNCTION	Definition
FX(X)	The objective function.
DELFX(NX, X)	The partial derivatives of the objective function.
GIX(NGI, X)	The non-linear constraint functions.
DELGIX(NX,NGI,X)	The partial derivatives of the non-linear constraints.

The arguments of the FUNCTIONS statements are:

Argument	Definition
X	The vector $\underline{x}$.
NX	Subscript of the partial derivative desired, i.e. $\partial F(\underline{x})/\partial x_{NX}$.
NGI	The non-linear constraint number.

Each routine must include a statement dimensioning X(e.g.; REAL*4 X(8), or DIMENSION X(15)), at least one assignment statement with the name of the FUNCTION on the left of the equal sign (e.g.; FX = X(1), or DELGIX = 4.0 * X(2)**3), and at least one RETURN statement. No variation in the names of the FUNCTIONs, nor in the order of the arguments is allowed. The following example illustrates the use of the FUNCTION routines.

$$\text{Max:} \quad F(\underline{x}) = 6x_1 + 8x_2 - x_1^2 - x_2^2$$

$$\text{Subj to:} \quad \left.\begin{array}{l} 4x_1^2 + x_2^2 \leq 16 \\ (4 - x_1)^2 + x_2^2 \leq 16 \end{array}\right\} \text{non-linear constraints}$$

$$\left.\begin{array}{l} 3x_1 + 5x_2 \leq 15 \\ 2x_1 + 12x_2 \leq 30 \end{array}\right\} \text{linear constraints}$$

$$0 \leq x_1 \leq 1.85$$

$$2.0 \leq x_2.$$

For this problem the following FUNCTION routines are required:

a. The objective function

```
      FUNCTION FX( X )
      REAL*4 X(2)
      FX = 6.0 * X(1) + 8.0 * X(2) - X(1)**2 - X(2)**2
      RETURN
      END
```

b. The partial derivatives of the objective function

```
      FUNCTION DELFX( NX, X )
      REAL*4 X(2)
      GO TO ( 101, 102 ), NX
  101 DELFX = 6.0 - 2.0 * X(1)
      RETURN
  102 DELFX = 8.0 - 2.0 * X(2)
      RETURN
      END
```

c. The non-linear constraint functions

```
      FUNCTION GIX( NGI, X )
      REAL*4 X(2)
      GO TO ( 100, 200 ), NGI
  100 GIX = 4.0 * X(1)**2 + X(2)**2
      RETURN
  200 GIX = ( 4.0 - X(1) )**2 + X(2)**2
      RETURN
      END
```

d. The partial derivatives of the non-linear constraints

```
      FUNCTION DELGIX( NX, NGI, X )
      REAL*4 X(2)
      GO TO ( 100, 200 ), NGI
  100 GO TO ( 101, 102 ), NX
  101 DELGIX = 8.0 * X(1)
      RETURN
  102 DELGIX = 2.0 * X(2)
```

```
      RETURN
  200 GO TO ( 201, 202 ), NX
  201 DELGIX = 2.0 * X(1) - 8.0
      RETURN
  202 DELGIX = 2.0 * X(2)
      RETURN
      END
```

The significant feature to note is that when a FUNCTION is called by FEASBL specifying a point X, and when necessary the partial derivative desired NX, and the constraint number NGI, the appropriate value is returned.

2. Data Input

The remainder of the data needed is read in from data cards at execution in NAMELIST format. This format was chosen to minimize keypunch errors, and provide flexibility for the user. The following variables are required:

Name	Description
IXMAX	Number of variables.
IGCNT	Number of non-linear constraints.
ILCNT	Number of linear constraints.

The following variables are optional, and if not provided by the user will take on the default values listed.

Name	Default	Description
GL(I,J)	0.0	Linear constraint matrix element
XMIN(I)	0.0	Lower bound on x_j
XMAX(I)	$\simeq 10^{75}$	Upper bound on x_i
BC(I)	0.0	The b_i for the non-linear constraints.

Name	Default	Description
BL(I)	0.0	The b_i for the linear constraints.
X(I)	0.0	The initial feasible point.
EPSLON	0.1	The initial anti-jamming constant.
STPMIN	10^{-3}	The minimum step length.
PERCNT	10^{-2}	The minimum objective function improvement.
ITMAX	25	The maximum number of iterations.

The first data card begins with an ampersand (&) in column two followed by the word INPUT with no space (&INPUT). The values are then listed as "variable name = FORTRAN constant" i.e. IXMAX = 2, GL(1,1) = 3.0, the last data item is followed by an ampersand and the word "END" (&END). If more then one card is required, each card must begin in column 2 or more, and the &END will appear on the last card only. For the above example the data cards would look like:

```
&INPUT IXMAX = 2, IGCNT=2, ILCNT=2, GL(1,1)=3.0, GL(1,2)=5.0,
GL(2,1)=2.0, GL(2,2)=12.0, XMAX(1)=1.85, XMIN(2)=2.0,
BC(1)=16.0, BC(2)=16.0, BL(1)=15.0, BL(2)=30.0, ITMAX=10,
EPSLON=0.01, STPMIN=1.0E-04, X(1)=1.0, X(2)=2.0, &END
```

Here the order is unimportant, and all the variables need not be listed. In this example the default values for XMIN(1), XMAX(2), and PERCNT will be used since they are not listed on the data cards.

3. Data Limits

Input errors will result if the variable inputs are outside the following limits:

$1 \leq IXMAX \leq 50$

$0 \leq IGCNT \leq 50$

$0 \leq ILCNT \leq 50$

$XMIN(I) < XMAX(I)$

To prevent stopping prematurely, EPSLON should be significantly larger than STPMIN.

After all the data is input, the initial point is checked for feasibility. If the point is not feasible the program is terminated.

4. The Program CHECK

FEASBL requires 290K of core space in addition to that required for the FUNCTION routines. Most problems will require less then 320K and less then 4 minutes to run. As an aid for checking the input data and FUNCTION routines, the program CHECK can be used. CHECK requires exactly the same input as FEASBL, and performs the same input error checks. In addition CHECK will determine the feasibility of the initial point and print out the value of the objective function, all the constraints, and the value of all the partial derivatives of the objective function and non-linear constraints. By using CHECK the user can quickly verify the input, since CHECK can be run normally as a class B or C job, before using FEASBL, which is a class J job.

5. System Cards

The JCL cards required to use FEASBL or CHECK depend on the way that the FUNCTION routines are input. The routines

can be input as either Fortran source decks and compiled as part of the job, or the routines can be precompiled and input as object decks. An example of each method follows.

a. FEASBL with Source Decks

```
//   standard OS JOB card          ,TIME=(03,59)
// EXEC FORTCLG,REGION.GO=300K
//FORT.SYSIN DD *
       (FUNCTION source decks)
 /*
//LINK.NLPLIB DD DSNAME=F0600.NLPLIB,UNIT=2321,
//   VOL=SER=CEL001,DISP=SHR,LABEL=(,,,IN)
//LINK.SYSIN DD *
    INCLUDE NLPLIB(FEASBL,DARRAY)
 /*
//GO.SYSIN DD *
 &INPUT IXMAX=2, IGCNT=2, ... , &END
 /*
```

b. FEASBL with Object Decks

```
//    standard OS JOB card       ,TIME=(03,59)
// EXEC LGO,REGION.GO=300K
//LINK.NLPLIB DD DSNAME=F0600.NLPLIB,UNIT=2321,
//   VOL=SER=CEL001,DISP-SHR,LABEL=(,,,IN)
//LINK.SYSIN DD *
    INCLUDE NLPLIB(FEASBL,DARRAY)
       (FUNCTION object decks)
/*
//GO.SYSIN DD *
 &INPUT IXMAX=2, IGCNT=2, ... , X(2)=2.0, &END
/*
```

c. CHECK with Source Decks

```
//    standard OS JOB card
// EXEC FORTCLG,REGION.GO=100K
//FORT.SYSIN DD *
       (FUNCTION source decks)
/*
//LINK.NLPLIB DD DSNAME=F0600.NLPLIB,UNIT=2321,
//  VOL=SER=CEL001,DISP=SHR,LABEL=(,,,IN)
//LINK.SYSIN DD *
    INCLUDE NLPLIB(CHECK)
/*
//GO.SYSIN DD *
 &INPUT IXMAX=2, IGCNT=2, ... ,X(2)=2.0, &END
/*
```

d. CHECK with Object Decks

```
//      standard OS JOB card
// EXEC LGO,REGION.GO=100K
//LINK.NLPLIB DD DSNAME=F0600.NLPLIB,UNIT=2321,
//  VOL=SER=CEL001,DISP=SHR,LABEL=(,,,IN)
//LINK.SYSIN DD *
     INCLUDE NLPLIB(CHECK)
          (FUNCTION objective decks)
/*
//GO.SYSIN DD *
 &INPUT IXMAX=2, IGCNT=2, ... , X(2)=2.0, &END
/*
```

A set of object decks can be produced either after checking the FUNCTION routines by executing a FORTCD job, or in conjunction with the program CHECK by replacing // EXEC FORTCLG,REGION.GO=100K with // EXEC FORTCLGP,PARM.FORT='DECK',REGION.GO=100K.

B. SAMPLE PROGRAM

The FUNCTION routines, input cards, and system cards required to use FEASBL with source decks are shown for the problem:

$$\text{Max:} \quad F(\underline{x}) = 6x_1 + 8x_2 - x_1^2 - x_2^2$$

$$\text{Subj to:} \quad 4x_1^2 + x_2^2 \leq 16$$

$$(4 - x_1)^2 + x_2^2 \leq 16$$

$$3x_1 + 5x_2 \leq 15$$

$$2x_1 + 12x_2 \leq 30$$

$$0 \leq x_1 \leq 1.85$$

$$2.0 \leq x_2 \leq \infty$$

A sample of the output of both CHECK and FEASBL for this problem is provided. The next two pages are the cards required for executing FEASBL with the sample problem. This is followed by the output of CHECK and then FEASBL for the sample problem.

```
//SAMPLE01 JOB (1115,0601,ROY2),'SAMPLE JOB',TIME=(03,59)
// EXEC FORTCLG,REGION.GO=300K
//FORT.SYSIN DD *
      FUNCTION FX( X )
C
CC          THIS FUNCTION PROVIDES THE OBJECTIVE FUNCTION.
C
      REAL*4 X(2)
C
CC          THIS IS THE OBJECTIVE FUNCTION
C
      FX = 6.0 * X(1) + 8.0 * X(2) - X(1)**2 - X(2)**2
      RETURN
      END
C
CCCC                  ****************************************
C
      FUNCTION DELFX( NX, X )
C
CC    THIS FUNCTION PROVIDES THE FIRST PARTIAL DERIVATIVES
CC    OF THE OBJECTIVE FUNCTION.
C
      REAL*4 X(2)
      GO TO ( 101, 102 ), NX
C
CC          THIS IS THE PARTIAL WITH RESPECT TO X(1).
C
  101 DELFX = 6.0 - 2.0 * X(1)
      RETURN
C
CC          THIS IS THE PARTIAL WITH RESPECT TO X(2)
C
  102 DELFX = 8.0 - 2.0 * X(2)
      RETURN
      END
C
CCCC                  ****************************************
C
      FUNCTION GIX( NGI, X )
C
CC          THIS FUNCTION PROVIDES THE NON-LINEAR CONSTRAINTS.
C
      REAL*4 X(2)
C
CC          GO TO THE PROPER CONSTRAINT.
C
      GO TO ( 100, 200 ), NGI
C
CC          THIS IS NON-LINEAR CONSTRAINT NUMBER 1.
C
  100 GIX = 4.0 * X(1)**2 + X(2)**2
      RETURN
C
CC          THIS IS NON-LINEAR CONSTRAINT NUMBER 2.
C
  200 GIX = ( 4.0 - X(1) )**2 + X(2)**2
      RETURN
      END
C
CCCC                  ****************************************
C
      FUNCTION DELGIX( NX, NGI, X )
C
CC    THIS FUNCTION PROVIDES THE FIRST PARTIAL DERIVATIVES
CC    OF THE NON-LINEAR CONSTRAINT FUNCTIONS.
C
```

```
      REAL*4 X(2)
C
CC         GO TO THE CONSTRAINT SPECIFIED.
C
      GO TO ( 100, 200 ),NGI
C
CC    THIS IS CONSTRAINT NUMBER ONE, WHICH PARTIAL IS WANTED
C
  100 GO TO ( 101, 102 ), NX
C
CC         THIS IS THE PARTIAL WITH RESPECT TO X(1).
C
  101 DELGIX = 8.0 * X(1)
      RETURN
C
CC         THIS IS THE PARTIAL WITH RESPECT TO X(2)
C
  102 DELGIX = 2.0 * X(2)
      RETURN
C
CC    THIS IS CONSTRAINT NUMBER TWO, WHICH PARTIAL IS WANTED
C
  200 GO TO ( 201, 202 ), NX
C
CC         THIS IS THE PARTIAL WITH RESPECT TO X(1)
C
  201 DELGIX = 2.0 * X(1) - 8.0
      RETURN
C
CC    THIS IS THE PARTIAL WITH RESPECT TO X(2)
C
  202 DELGIX = 2.0 * X(2)
      RETURN
      END

//LINK.NLPLIB DD DSN=F0600.NLPLIB,UNIT=2321,
//         VOL=SER=CEL001,DISP=SHR,LABEL=(,,,IN)
//LINK.SYSIN DD *
    INCLUDE NLPLIB(FEASBL,DARRAY)
//GO.SYSIN DD *
 &INPUT IXMAX=2, IGCNT=2, ILCNT=2, GL(1,1)=3.0, GL(1,2)=5.0,
 GL(2,1)=2.0, GL(2,2)=12.0, XMAX(1)=1.85, XMIN(2)=2.0,
 BC(1)=16.0, BC(2)=16.0, BL(1)=15.0, BL(2)=30.0, ITMAX=10,
 EPSLON=0.01, STPMIN=1.0E-04, X(1)=1.0, X(2)=2.0, &END
```

SAMPLE OUTPUT FROM CHECK

```
                                        NON-LINEAR PROGRAMMING PROBLEM

 INITIAL SET UP:
    NUMBER OF VARIABLES:                 2
    NUMBER OF NON-LINEAR CONSTRAINTS:    2
    NUMBER OF LINEAR CONSTRAINTS:        2
    THE LINEAR CONSTRAINT MATRIX IS:
    ( 1, 1):    3.00000         ( 1, 2):    5.00000

    ( 2, 1):    2.00000         ( 2, 2):    12.0000

    THE BOUNDS ON THE VARIABLES ARE:

       0.0          < X( 1) <     1.85000    ;            2.00000     < X( 2) <    0.723701E 7o
    NON-LINEAR CONSTRAINTS REQUIREMENTS VECTOR, BC(I):
    ( 1):           16.0000    ( 2):           16.0000    (
    LINEAR CONSTRAINTS REQUIREMENTS VECTOR, BL(I):
    ( 1):           15.0000    ( 2):           30.0000    (
    THE INITIAL FEASIBLE STARTING POINT IS:
         X( 1):          1.00000    ,X( 2):          2.00000    ,X(
    ANTI JAMMING CONSTANT:          0.100000E-01
    MINIMUM  STEP  LENGTH:          0.100000E-03
    MINIMUM   IMPROVEMENT:          0.100000E-01
    MAXIMUM  # ITERATIONS:                    10
INPUT COMPLETE.  INITIAL VALUE OF OBJECTIVE FUNCTION:   1.700000E 01

VALUE OF NON-LINEAR CONSTRAINT NUMBER,  1, IS:  8.0000000E 00
VALUE OF NON-LINEAR CONSTRAINT NUMBER,  2, IS:  1.3000000E 01
THE VALUE OF  LINEAR CONSTRAINT NUMBER  1, IS:   1.300000E 01
THE VALUE OF  LINEAR CONSTRAINT NUMBER  2, IS:   2.600000E 01
THE BOUNDS, AND ALL THE NON-LINEAR AND LINEAR CONSTRAINTS HAVE BEEN CHECKED.
NEXT THE VALUES OF THE PARTIAL DERIVATIVES OF THE OBJECTIVE FUNCTION AND THE CONSTRAINTS ARE LISTED.
THE VALUE OF DELFX( 1, X) IS:   4.000000E 00
THE VALUE OF DELFX( 2, X) IS:   4.000000E 00
THE VALUE OF DELGIX( 1, 1, X) IS:   8.000000E 00
THE VALUE OF DELGIX( 2, 1, X) IS:   4.000000E 00
THE VALUE OF DELGIX( 1, 2, X) IS:  -6.000000E 00
THE VALUE OF DELGIX( 2, 2, X) IS:   4.000000E 00
```

SAMPLE OUTPUT FROM FEASBL

```
                                        NON-LINEAR RROGRAMMING PROBLEM

INITIAL SET UP:
   NUMBER OF VARIABLES:                 2
   NUMBER OF NON-LINEAR CONSTRAINTS:    2
   NUMBER OF LINEAR CONSTRAINTS:        2
   THE LINEAR CONSTRAINT MATRIX IS:
   ( 1, 1):    3.00000          ( 1, 2):    5.00000

   ( 2, 1):    2.00000          ( 2, 2):    I2.0000

   THE BOUNDS ON THE VARIABLES ARE:

      0.0          < X( 1) <      1.85000     ;            2.00000      < X( 2) <    0.723701E 76
   NON-LINEAR CONSTRAINTS REQUIREMENTS VECTOR, BC(I):
   ( 1):           16.0000     ( 2):           16.0000     (
   LINEAR CONSTRAINTS REQUIREMENTS VECTOR, BL(I):
   ( 1):           15.0000     ( 2):           30.0000     (
   THE INITIAL FEASIBLE STARTING ROINT IS:
        X( 1):          1.00000     ,X( 2):          2.00000     ,X(
   ANTI JAMMING CONSTANT:          0.100000E-01
   MINIMUM  STER  LENGTH:          0.100000E-03
   MINIMUM   IMPROVEMENT:          0.100000E-01
   MAXIMUM  # ITERATIONS:                    10
INPUT COMPLETE.  INITIAL VALUE OF OBJECTIVE FUNCTION:   1.700000E 01


                                                        PROBLEM SOLUTION

ITERATION NUMBER:      1
    F(X) =    1.887500E 01;  IMPROVEMENT THIS ITERATION:   11.0294 RERCENT.
    THE ROINT X( ) IS:
    X( I),    1.250000    ; X( 2),    2.250000    ; X(
ITERATION NUMBER:      2
    F(X) =    1.922217E 01;  IMPROVEMENT THIS ITERATION:    1.8393 PERCENT.
    THE POINT X( ) IS:
    X( 1),    1.666625    ; X( 2),    2.000024    ; X(
ORTIMAL SOLUTION:
THE ORTIMAL VALUE OF THE OBJECTIVE FUNCTION IS:   1.922217E 01
    THE POINT X( ) IS:
    X( 1),    1.666625    ; X( 2),    2.000024    ; X(
```

V. EVALUATION

To validate the code, and evaluate its performance, several non-linear problems were solved with FEASBL. These problems were selected to represent the range of problems FEASBL was designed to solve. The problems included unconstrained optimization of a concave objective function, constrained optimization of concave objective functions some with the unconstrained maxima interior and some exterior to the feasible regions. An unbounded problem was also solved.

In each case the optimum solution was obtained in less then five seconds of cpu time. These problems were all small in order to facilitate manual verification of the solutions. Larger problems will take correspondingly more time, but the code appears to be quite efficient. In most problems the optimal solution was obtained after two iterations. In one case, the solution took seven iterations.

It was found that the value of STPMIN is critical to the performance of FEASBL. If STPMIN is set too small, when the interval between λ_F and λ_{NF} is calculated, the round-off error may exceed the value of STPMIN. This results in the code looping indefinitely trying to reduce the interval beyond machine accuracy leading to a program completion due to excess time. If this should occur the problem should be restarted at the last point with a larger value for STPMIN. With step lengths on the order of 1.0 the interval cannot be

determined to an accuracy greater then 10^{-6}. With larger step lengths this value will increase. An initial value of 10^{-2} or 10^{-3} is recommended for STPMIN. This value can be reduced in subsequent runs of FEASBL if a more precise solution is desired.

The initial value of EPSLON is not as critical as STPMIN. EPSLON should be greater than STPMIN but generally not greater than 10^{-1}. Experience with FEASBL indicates that values on the order of 10^{-1} or 10^{-2} perform well. If a large value is used, on the order of 1.0 or greater, and an optimal point is found, this should be verified using a smaller value for EPSLON, as the code may stop early with large values of EPSLON.

Performance was uneffected by variations in PERCNT. Large values may cause the program to terminate sooner. Values less then 10^{-6} are equivalent to zero because of inherent machine accuracy.

An attempt was made to solve a problem in which all the constraints were equality constraints, thus the feasible region had no strictly interior points. The program would terminate immediately without any significant movement. This phenomenon was anticipated since the underlying theory assumes that the feasible region contains at least one strictly interior point to the set of non-linear constraints [1].

```
C**********************************************************************INT00010
C**********************************************************************INT00020
C                                                                      INT00030
C                          PROGRAM FEASBL                              INT00040
C                                                                      INT00050
C     PURPOSE:  TO MAXIMAZE A NON-LINEAR OBJECTIVE FUNCTION, OVER      INT00060
C     A CONVEX SET OF CONSTRAINTS.  THE MAXIMAZATION IS DONE BY        INT00070
C     THE METHOD OF FEASIBLE DIRECTIONS PROPOSED BY ZOUTENDIJK.        INT00080
C                                                                      INT00090
C     USAGE:                                                           INT00100
C                                                                      INT00110
C     THE USER MUST PROVIDE THE FOLLOWING FORTRAN "FUNCTION"           INT00120
C     ROUTINES:                                                        INT00130
C                                                                      INT00140
C               NAME:                 DEFINITION                     INT00150
C                                                                      INT00160
C               FX( X )               THE OBJECTIVE FUNCTION F(X)    INT00170
C                                                                      INT00180
C               DELFX( NX, X )        THE FIRST PARTIAL DERIVATIVES OF INT00190
C                                     THE OBJECTIVE FUNCTION.          INT00200
C                                                                      INT00210
C               GIX( NGI, X )         NON-LINEAR CONSTRAINT FUNCTIONS. INT00220
C                                                                      INT00230
C               DELGIX(NX, NGI, X)    THE FIRST PARTIAL DERRIVATIVES OFINT00240
C                                     THE NON-LINEAR CONSTRAINTS.      INT00250
C                                                                      INT00260
C     THE ARGUMENTS OF THE FUNCTIONS ARE:                              INT00270
C                                                                      INT00280
C               NAME                  DEFINITION                     INT00290
C                                                                      INT00300
C               X                     THE VECTOR X                   INT00310
C                                                                      INT00320
C               NX                    PARTIAL DERIVATIVE DESIRED,    INT00330
C                                     I.E. THE PARTIAL WITH RESPECT    INT00340
C                                     TO X(NX).                        INT00350
C                                                                      INT00360
C               NGI                   NON-LINEAR CONSTRAINT NUMBER.  INT00370
C                                                                      INT00380
C     THE USER MUST PROVIDE THE FOLLOWING DATA IN NAMELIST FORMAT:     INT00390
C                                                                      INT00400
C     THE FOLLOWING VARIABLES ARE REQUIRED:                            INT00410
C                                                                      INT00420
C               NAME                  DEFINITION                     INT00430
C                                                                      INT00440
C               IXMAX                 NUMBER OF VARIABLES.           INT00450
C               IGCNT                 NUMBER OF NON-LINEAR CONSTRAINTS.INT00460
C                                                                      INT00470
C               ILCNT                 NUMBER OF LINEAR CONSTRAINTS.  INT00480
```

```
C                                                                       INT00490
C     THE FOLLOWING VARIABLES ARE OPTIONAL:                             INT00500
C                                                                       INT00510
C               NAME                    DEFINITION                      INT00520
C                                                                       INT00530
C               GL(I,J)                 LINEAR CONSTRAINT MATRIX ELEMENT.INT00540
C                                                                       INT00550
C               XMIN(I)                 LOWER BOUND ON X(I).            INT00560
C                                                                       INT00570
C               XMAX(I)                 UPPER BOUND ON X(I).            INT00580
C                                                                       INT00590
C               BC(I)                   NON-LINEAR CONSTRAINT           INT00600
C                                       REQUIREMENT VECTOR ELEMENT.     INT00610
C                                                                       INT00620
C               BL(I)                   LINEAR CONSTRAINT REQUIREMENTS  INT00630
C                                       VECTOR ELEMENT.                 INT00640
C                                                                       INT00650
C               X(I)                    THE INITIAL FEASIBLE POINT.     INT00660
C                                                                       INT00670
C               EPSLON                  THE INITIAL ANTI-JAMMING        INT00680
C                                       CONSTANT.                       INT00690
C                                                                       INT00700
C               STPMIN                  THE MINIMUM STEP LENGTH.        INT00710
C                                                                       INT00720
C               PERCNT                  THE MINIMUM OBJECTIVE FUNCTION  INT00730
C                                       IMPROVEMENT.                    INT00740
C                                                                       INT00750
C               ITMAX                   MAXIMUM NUMBER OF ITERATIONS.   INT00760
C                                                                       INT00770
C     REMARKS:                                                          INT00780
C                                                                       INT00790
C     FOR A COMPLETE DESCRIPTION SEE;                                   INT00800
C                                                                       INT00810
C     "A COMPUTER CODE FOR SOLVING MEDIUM SIZED NON-LINEAR              INT00820
C     PROGRAMMING PROBLEMS BY THE METHOD OF FEASIBLE DIRECTIONS"        INT00830
C     BY LCDR J. D. HARRISON, SEPTEMBER 1974.                           INT00840
C                                                                       INT00850
C                                                                       INT00860
      IMPLICIT REAL*4 (A-H,O-Z), INTEGER*2 (I-L), INTEGER*4 (M-N),      FEAS0010
     ************************************************************       INT00870
C                                                                       INT00880
     1LOGICAL*1 ($)                                                     FEAS0020
      REAL*8  BINVRS(152,152), DSUM, DZJ, DBI, DEL                      FEAS0030
      DIMENSION ACORE(101,100), AJ(152), B(100), BC(50), BL(50),PHI(152)FEAS0040
      DIMENSION GLCNST(50,50), X(50), XB(152), XMAX(50), XMIN(50)       FEAS0050
      DIMENSION XONE(50), XSAVE(50), XTWO(50), S(50), YJ(152), ZJCJ(253)FEAS0060
      DIMENSION IAJ(100), IBASIC(152), IGBIND(50), ILBIND(50), IMAX(50) FEAS0070
      DIMENSION IMIN(50), IXIN(50), NWORKL(152), NWORKM(152)            FEAS0080
```

```
      DIMENSION $BASIC(253), $YJPOS(253)                                FEAS0090
      DIMENSION GL(50,50)                                               FEAS0095
      EQUIVALENCE    (B(1), BC(1) ), ( B(51), BL(1) )                   FEAS0100
      EQUIVALENCE ( YJ(1), PHI(1) ), ( XB(1), XDIR )                    FEAS0110
      EQUIVALENCE ( YJ(1), NWORKL(1) ), ( AJ(1), NWORKM(1) )            FEAS0120
      EQUIVALENCE ( YJ(1), XSAVE(1) ), (YJ(51), XONE(1) )               FEAS0130
      EQUIVALENCE ( YJ(147), XLAMUP ), ( YJ(148), XLAMNF )              FEAS0140
      EQUIVALENCE ( YJ(149), XLAMF ), ( YJ(150), XLAMDA )               FEAS0150
      EQUIVALENCE ( YJ(151), XINC ), ( AJ(1), XTWO(1) )                 FEAS0160
      EQUIVALENCE ( GL(1,1), GLCNST(1,1) )                              FEAS0165
      NAMELIST / INPUT / IXMAX, IGCNT, ILCNT, GL, XMIN, XMAX,           FEAS0166
     1 BC, BL, X, EPSLON, STPMIN, PERCNT, ITMAX                         FEAS0167
C                                                                       FEAS0170
CC         SET THE INITIAL VALUES                                       FEAS0180
C                                                                       FEAS0190
      DATA BINVRS / 23104 * 0.0D0 /                                     FEAS0200
      DATA ZJCJ / 253*0.0E0 /, B / 100*0.0E0 /, YJ / 152*0.0E0 /        FEAS0210
      DATA XMIN / 50*0.0E0 /, XMAX / 50*Z7FFFFFFF /, ITER / 1 /         FEAS0220
      DATA EPSLON / 1.0E-01 /,STPMIN / 1.0E-03 /, PERCNT / 1.0E-02 /    FEAS0230
      DATA CMAX / Z7FFFFFFF /,NREAD / 5 /, NWRITE / 6 /                 FEAS0240
      DATA ZCJTST /-1.0E-05/                                            FEAS0244
      DATA GLCNST / 2500*0.0E0 /,  ITMAX / 25 /                         FEAS0245
      DATA X / 50*0.0E0 /                                               FEAS0246
C                                                                       FEAS0250
CCC   INPUT SECTION:                                                    FEAS0260
CCCCC      READ IN FOLLOWING VALUES: IXMAX, IGCNT, ILCNT, GLCNST(I,J),  FEAS0270
CCCCC      XMIN(I), XMAX(I), BC(I), BL(I), X(I), EPSLON, STPMIN, PERCNT  FEAS0280
CCCCC      PRINT OUT ALL INPUT VALUES AND INITIAL VALUE OF OBJECT2VE     FEAS0290
CCC        FUNCTION.                                                    FEAS0300
C                                                                       FEAS0310
      READ(NREAD,INPUT)                                                 FEAS0320
      WRITE(NWRITE,901) IXMAX, IGCNT, ILCNT                             FEAS0340
  901 FORMAT('1',50X,'NON-LINEAR PROGRAMMING PROBLEM',///,'0  INITIAL SEFEAS0350
     1T UP:',/,'0',5X,'NUMBER OF VARIABLES:',13X,I5,/,'0',5X,'NUMBER OF FEAS0360
     2NON-LINEAR CONSTRAINTS:',I5,/,'0',5X,'NUMBER OF LINEAR CONSTRAINTSFEAS0370
     3:',4X,I5)                                                         FEAS0380
C                                                                       FEAS0385
CCC                  VERIFY THAT THE PARAMETERS ARE WITHIN LIMITS       FEAS0390
C                                                                       FEAS0395
      IF ( IXMAX .LE. 50 .AND. IGCNT .LE. 50 .AND. ILCNT .LE. 50 ) GO TOFEAS0400
     1 1000                                                             FEAS0410
      WRITE(NWRITE,902)                                                 FEAS0420
  902 FORMAT('0 * * * * * * * ERROR IN ABOVE PARAMETERS, PROGRAM TERMINATEFEAS0430
     1D * * * * * * *')                                                 FEAS0440
      STOP                                                              FEAS0450
 1000 IF( ILCNT .EQ. 0 ) GO TO 1005                                     FEAS0460
      WRITE(NWRITE,904)                                                 FEAS0490
  904 FORMAT('0',5X,'THE LINEAR CONSTRAINT MATRIX IS:')                 FEAS0500
```

```
      DO 1004 I=1,ILCNT                                                 FEAS0510
          DO 1003 N=1,IXMAX                                             FEAS0520
          IF( MOD(N,3) .EQ. 1 ) WRITE(NWRITE,1902) I, N, GL(I,N)        FEAS0530
          IF( MOD(N,3) .EQ. 2 ) WRITE(NWRITE,1903) I, N, GL(I,N)        FEAS0540
          IF( MOD(N,3) .EQ. 0 ) WRITE(NWRITE,1904) I, N, GL(I,N)        FEAS0550
 1902     FORMAT('0',05X,'(',I2,',',I2,');',G15.6)                      FEAS0570
 1903     FORMAT('+',33X,'(',I2,',',I2,');',G15.6)                      FEAS0580
 1904     FORMAT('+',61X,'(',I2,',',I2,');',G15.6)                      FEAS0590
 1003     CONTINUE                                                      FEAS0672
          WRITE(NWRITE,1901)                                            FEAS0674
 1901     FORMAT('0')                                                   FEAS0676
 1004 CONTINUE                                                          FEAS0680
 1909 FORMAT('0',5X,'THE BOUNDS ON THE VARIABLES ARE:',//)              FEAS0760
 1005 WRITE(NWRITE,1909)                                                FEAS0750
      DO 1018 N=1,IXMAX                                                 FEAS0770
          IF( MOD(N,2) .EQ. 1 ) WRITE(NWRITE,1912) XMIN(N), N, XMAX(N)  FEAS0790
 1912     FORMAT(6X,G15.6,' < X(',I2,') < ',G15.6,';')                  FEAS0800
          IF( MOD(N,2) .EQ. 0 ) WRITE(NWRITE,1913) XMIN(N), N, XMAX(N)  FEAS0810
 1913     FORMAT('+',55X,G15.6,' < X(',I2,') < ',G15.6)                 FEAS0820
 1018 CONTINUE                                                          FEAS0830
C                                                                       FEAS0835
CCCC                  CHECK THE UPPER AND LOWER BOUNDS FOR OBVIOUS      FEAS0836
CCCC                  ERRORS.  STOP IF ANY EXIST.                       FEAS0837
C                                                                       FEAS0838
      DO 1019 I=1,IXMAX                                                 FEAS0840
          IF(XMIN(I) .LT. XMAX(I) ) GO TO 1019                          FEAS0850
 1919 FORMAT('0 * * * * * ERROR IN BOUNDS, PROGRAM TERMINATED. * * * *')FEAS0860
      STOP                                                              FEAS0870
 1019 CONTINUE                                                          FEAS0880
 1020 IF( IGCNT .EQ. 0 ) GO TO 1030                                     FEAS0890
      WRITE(NWRITE,1921) (I,BC(I), I=1,IGCNT)                           FEAS0920
 1921 FORMAT('0',5X,'NON-LINEAR CONSTRAINTS REQUIREMENTS VECTOR, BC(I):'FEAS0930
     1,/,('0',5X,4('(',I2,');  ',G20.6)/))                              FEAS0940
 1030 IF( ILCNT .EQ. 0 ) GO TO 1035                                     FEAS0950
      WRITE(NWRITE,1930) (I, BL(I), I=1, ILCNT)                         FEAS0970
 1930 FORMAT('0',5X,'LINEAR CONSTRAINTS REQUIREMENTS VECTOR, BL(I):',/, FEAS0980
     1('0',5X,4('(',I2,');  ',G20.6)/))                                 FEAS0990
 1035 WRITE(NWRITE,1935) ( I, X(I), I=1, IXMAX )                        FEAS1010
 1935 FORMAT('0',5X,'THE INITIAL FEASIBLE STARTING POINT IS:',/,        FEAS1020
     1('0',10X,4('X(',I2,'); ',G20.6,',')/))                            FEAS1030
      WRITE(NWRITE,1937) EPSLON, STPMIN, PERCNT, ITMAX                  FEAS1060
 1937 FORMAT('0',5X,'ANTI JAMMING CONSTANT:  ',G20.6,/,'0',5X,'MINIMUM  FEAS1070
     1STEP  LENGTH:  ',G20.6,/,'0',5X,'MINIMUM   IMPROVEMENT:  ',G20.6, FEAS1080
     2/,'0',5X,'MAXIMUM  # ITERATIONS:  ',G20.6)                        FEAS1090
      FOFX = FX(X)                                                      FEAS1100
      WRITE(NWRITE,1938) FOFX                                           FEAS1110
 1938 FORMAT('0 INPUT COMPLETE.  INITIAL VALUE OF OBJECTIVE FUNCTION:', FEAS1120
     1PE15.6,/'1',58X,'PROBLEM SOLUTION')                               FEAS1130
```

```
C                                                                       FEAS1140
C          DETERMINE THE BINDING CONSTRAINTS.                           FEAS1150
C          CHECK THE UPPER AND LOWER BOUNDS, NON-LINEAR, AND LINEAR     FEAS1160
C          CONSTRAINTS.  RECORD THOSE THAT ARE BINDING, OR ARE WITHIN   FEAS1170
C          EPSLON OF BINDING.  IF THE POINT IS NOT FEASIBLE STOP.       FEAS1180
C                                                                       FEAS1190
C          INITIALIZE THE COUNTERS                                      FEAS1200
C                                                                       FEAS1205
 2000 KOUNT1 = 0                                                        FEAS1210
      KOUNT2 = 0                                                        FEAS1220
      KOUNT3 = 0                                                        FEAS1230
C                                                                       FEAS1235
C          CHECK THE UPPER AND LOWER BOUNDS ON X.                       FEAS1240
C                                                                       FEAS1245
      DO 2010 I=1,IXMAX                                                 FEAS1250
          XX = X(I)                                                     FEAS1260
          XLO = XMIN(I)                                                 FEAS1270
          XHI = XMAX(I)                                                 FEAS1280
C                                                                       FEAS1285
C                    IS X(I) NEAR ITS LOWER BOUND.                      FEAS1290
C                                                                       FEAS1295
          IF( XX .LE. XLO + EPSLON ) GO TO 2002                         FEAS1300
C                                                                       FEAS1305
C                    IS X(I) NEAR UPPER BOUND                           FEAS1310
C                                                                       FEAS1315
          IF( XX .LT. XHI - EPSLON ) GO TO 2004                         FEAS1320
C                                                                       FEAS1325
C                    IS X(I) ABOVE ITS UPPER BOUND                      FEAS1330
C                                                                       FEAS1335
          IF( XX .GT. XHI ) GO TO 2090                                  FEAS1340
C                                                                       FEAS1345
C                    X(I) IS WITHIN EPSLON OF UPPER BOUND               FEAS1350
C                                                                       FEAS1355
          KOUNT2 = KOUNT2 + 1                                           FEAS1360
          IMAX(KOUNT2) = I                                              FEAS1370
          GO TO 2010                                                    FEAS1380
C                                                                       FEAS1385
C                    X(I) IS INFEASIBLE, STOP                           FEAS1390
C                                                                       FEAS1395
 2090     WRITE(NWRITE,2990) I, XX                                      FEAS1400
 2990 FORMAT('0 * * * * * X(',I2,') = ',1PE20.7,', IS OUTSIDE ITS BOUNDSFEAS1410
     1, POINT IS NOT FEASIBLE.  PROGRAM TERMINATED. * * * * *')         FEAS1420
          STOP                                                          FEAS1430
C                                                                       FEAS1435
C                    IS X(I) BELOW LOWER BOUND                          FEAS1440
C                                                                       FEAS1445
 2002     IF(XX .LT. XLO ) GO TO 2090                                   FEAS1450
C                                                                       FEAS1455
```

```
C                   X(I) IS WITHIN EPSLON OF LOWER BOUND                    FEAS1460
C                                                                           FEAS1465
          KOUNT1 = KOUNT1 + 1                                               FEAS1470
          IMIN(KOUNT1) = I                                                  FEAS1480
          GO TO 2010                                                        FEAS1490
C                                                                           FEAS1495
C                   X(I) IS INTERIOR                                        FEAS1500
C                                                                           FEAS1505
 2004     KOUNT3 = KOUNT3 + 1                                               FEAS1510
          IXIN(KOUNT3) = I                                                  FEAS1520
 2010 CONTINUE                                                              FEAS1530
C                                                                           FEAS1535
C                   STORE NUMBER OF BINDING AND NONBINDING BOUNDS           FEAS1540
C                                                                           FEAS1545
      JLO = KOUNT1                                                          FEAS1550
      JHI = KOUNT2                                                          FEAS1560
      JIN = KOUNT3                                                          FEAS1570
C                                                                           FEAS1575
C                   ARE THERE ANY NON-LINEAR CONSTRAINTS                    FEAS1580
C                                                                           FEAS1585
      IF( IGCNT .LE. 0 ) GO TO 2030                                         FEAS1590
C                                                                           FEAS1595
C                   RESET THE COUNTER, THEN CHECK THE N.L. CONSTRAINTS      FEAS1600
C                                                                           FEAS1605
      KOUNT1 = 0                                                            FEAS1610
      DO 2020 N=1,IGCNT                                                     FEAS1620
          XX = GIX(N,X)                                                     FEAS1630
          XHI = BC(N)                                                       FEAS1640
          XLO = XHI - EPSLON                                                FEAS1650
C                                                                           FEAS1655
C                   IS X INTERIOR TO CONSTRAINT N                           FEAS1660
C                                                                           FEAS1665
          IF( XX .LT. XLO ) GO TO 2020                                      FEAS1670
C                                                                           FEAS1675
C                   IS X INFEASIBLE WITH RESPECT TO CONSTRINT N             FEAS1680
C                                                                           FEAS1685
          IF( XX .GT. XHI ) GO TO 2092                                      FEAS1690
C                                                                           FEAS1695
C                   X IS WITHIN EPSLON OF CONSTRAINT N                      FEAS1700
C                                                                           FEAS1705
          KOUNT1 = KOUNT1 + 1                                               FEAS1710
          IGBIND(KOUNT1) = N                                                FEAS1720
          GO TO 2020                                                        FEAS1730
 2092     WRITE(NWRITE,2992) N, XX                                          FEAS1740
 2992     FORMAT('0 * * * * * X IS NOT FEASIBLE WITH RESPECT TO NON-LINEFEAS1750
     1AR CONSTRAINT NUMBER',I3,' VALUE OF CONSTRAINT IS:',1PE20.7)          FEAS1760
          STOP                                                              FEAS1765
 2020 CONTINUE                                                              FEAS1770
```

```
C                                                                       FEAS1775
CC                    STORE NUMBER OF BINDING N. L. CONSTRAINTS.        FEAS1780
C                                                                       FEAS1785
      JNGI = KOUNT1                                                     FEAS1790
      GO TO 2032                                                        FEAS1800
C                                                                       FEAS1805
CC        IF WE GET HERE, THERE ARE NO NON-LINEAR CONSTRAINTS.          FEAS1810
C                                                                       FEAS1815
 2030 JNGI = 0                                                          FEAS1820
C                                                                       FEAS1825
CC                    ARE THERE ANY LINEAR CONSTRAINTS                  FEAS1830
C                                                                       FEAS1835
 2032 IF( ILCNT .LE. 0 ) GO TO 2042                                     FEAS1840
C                                                                       FEAS1845
CC                    RESET THE COUNTER, THEN CHECK THE LINEAR CONSTRAINTSFEAS1850
C                                                                       FEAS1855
      KOUNT1 = 0                                                        FEAS1860
      DO 2040 N=1,ILCNT                                                 FEAS1870
          DSUM = 0.0D0                                                  FEAS1880
C                                                                       FEAS1885
CC                    EVALUATE THE N-TH LINEAR CONSTRAINT               FEAS1890
C                                                                       FEAS1895
          DO  2035 I=1,IXMAX                                            FEAS1900
          DEL = DBLE( GLCNST(N,I) )                                     FEAS1910
          DBI = DBLE( X(I) )                                            FEAS1920
C                                                                       FEAS1925
CC                    CHECK FOR NEAR 0 VALUES TO PREVENT UNDERFLOW.     FEAS1930
C                                                                       FEAS1935
          IF( DABS(DBI) .LT. 1.0D-39 ) GO TO 2035                       FEAS1940
          IF( DABS(DEL) .LT. 1.0D-39 ) GO TO 2035                       FEAS1950
          DSUM = DSUM + DEL * DBI                                       FEAS1960
 2035     CONTINUE                                                      FEAS1970
          XHI = BL(N)                                                   FEAS1980
          XX = SNGL(DSUM)                                               FEAS1990
          XLO = XHI - EPSLON                                            FEAS2000
C                                                                       FEAS2005
CC                    IS X INTERIOR WITH RESPECT TO CONSTRAINT N        FEAS2010
C                                                                       FEAS2015
          IF( XX .LT. XLO ) GO TO 2040                                  FEAS2020
C                                                                       FEAS2025
CC                    IS X INFEASIBLE WITH RESPECT TO CONSTRAINT N      FEAS2030
C                                                                       FEAS2035
          IF( XX .GT. XHI ) GO TO 2094                                  FEAS2040
C                                                                       FFAS2045
CC                    X IS WITHIN EPSLON OF CONSTRAINT N                FEAS2050
C                                                                       FEAS2055
          KOUNT1 = KOUNT1 + 1                                           FEAS2060
          ILBIND(KOUNT1) = N                                            FEAS2070
```

```
          GO TO 2040                                                        FEAS2080
 2094     WRITE(NWRITE,2994) N,XX                                           FEAS2090
 2994     FORMAT('0 * * * * * X IS NOT FEASIBLE WITH RESPECT TO LINEAR CFEAS2100
     1ONSTRAINT NUMBER',I3,' VALUE OF CONSTRAINT IS:',1PE20.7)              FEAS2110
          STOP                                                              FEAS2120
 2040 CONTINUE                                                              FEAS2130
C                                                                           FEAS2135
C                    STORE NUMBER OF BINDING LINEAR CONSTRAINTS             FEAS2140
C                                                                           FEAS2145
      JNLI = KOUNT1                                                         FEAS2150
      GO TO 2100                                                            FEAS2160
C                                                                           FEAS2165
C                    THERE ARE NO LINEAR CONSTRAINTS                        FEAS2170
C                                                                           FEAS2175
 2042 JNLI = 0                                                              FEAS2180
C                                                                           FEAS2190
C                                                                           FEAS2295
C         SET UP THE INITIAL XB VECTOR AND BASIS INVERSE                    FEAS2200
C         VARIABLES:  ICROW - # OF ROWS IN THE "CORE" OF THE"A" MATRIX      FEAS2210
C                     ILPROW - # OF ROWS IN DIR. FINDING PROB. L P.         FEAS2220
C                     ICCOL - # OF COLUMNS IN "CORE"                        FEAS2230
C                     ILPCOL - # OF COLUMNS IN L. P.                        FEAS2240
C                     IBASIC(I) - LIST OF VARIABLES IN BASIS                FEAS2250
C                     $BASIC(I) - BASIS STATUS OF ALL VARIABLES             FEAS2260
C                     XB(I) - VALUE OF VECTORS IN BASIS                     FEAS2270
C                                                                           FEAS2280
C         SET VALUES OF ICROW, ICPROW, ICCOL, ILPCOL                        FEAS2290
 2100 ICROW = JNGI + JNLI + 2                                               FEAS2300
      ILPROW = ICROW + IXMAX                                                FEAS2310
      ICCOL = JLO + JHI + 2 * JIN + 2                                       FEAS2320
      ILPCOL = ICCOL + ILPROW - 1                                           FEAS2330
C                                                                           FEAS2335
C                    XB WILL BE OF FORM (0,0, ... ,0,1, ... ,1)             FEAS2340
C                                                                           FEAS2345
      DO 2102 I=1,ICROW                                                     FEAS2350
          XB(I) = 0.0E0                                                     FEAS2360
 2102 CONTINUE                                                              FEAS2370
      I1 = ICROW + 1                                                        FEAS2380
      IBASIC(1) = 1                                                         FEAS2390
      $BASIC(1) = .TRUE.                                                    FEAS2400
      DO 2104 I=I1,ILPCOL                                                   FEAS2410
          XB(I) = 1.0E0                                                     FEAS2420
 2104 CONTINUE                                                              FEAS2430
C                                                                           FEAS2435
C         THE BASIS INVERSE IS INITIALLY THE IDENTITY MATRIX                FEAS2440
C                                                                           FEAS2445
      DO 2108 I=1,ILPROW                                                    FEAS2450
          DO 2106 N=1,ILPROW                                                FEAS2460
```

```
                BINVRS(N,I) = 0.0D0                                     FEAS2470
C                                                                       FEAS2475
CC                    SET THE DIAGONAL ELEMENTS TO 1.0                  FEAS2480
C                                                                       FEAS2485
              IF( N .EQ. I ) BINVRS(I,I) = 1.0D0                        FEAS2490
 2106     CONTINUE                                                      FEAS2500
 2108 CONTINUE                                                          FEAS2510
C                                                                       FEAS2515
CC          THE ONLY VARIABLES IN THE BASIS ARE 1 AND THE SLACK VARIABLES FEAS2520
C                                                                       FEAS2525
      DO 2110 I=2,ICCOL                                                 FEAS2530
          $BASIC(I) = .FALSE.                                           FEAS2540
 2110 CONTINUE                                                          FEAS2550
      I1 = ICCOL + 1                                                    FEAS2560
      DO 2112 I=I1,ILPCOL                                               FEAS2570
          $BASIC(I) = .TRUE.                                            FEAS2580
          IBASIC(I-I1+2) = I                                            FEAS2590
 2112 CONTINUE                                                          FEAS2600
C                                                                       FEAS2610
CCCC        BUILD THE "A" MATRIX:                                       FEAS2620
CCCCC            WE WILL CALCULATE AND STORE THOSE ELEMENTS OF THE      FEAS2630
CCCCC       L. P. "A" MATRIX THAT DEPEND ON THE VALUE OF X, AND ON      FEAS2640
CCCCC       WHICH OF THE CONSTRAINTS.  THESE WILL BE STORED IN A MATRIX FEAS2650
CCCCC       CALLED "ACORE".  THE OTHER ELEMENTS ARE ALL EITHER 0.0 OR   FEAS2660
CCCCC       1.0 AND DO NOT DEPEND ON X, SO THEY WILL BE GENERATED ONLY  FEAS2670
CCCCC       WHEN NEEDED.                                                FEAS2680
CCCCC                                                                   FEAS2690
CCCCC                ARE THERE ANY VARIABLES AT THEIR LOWER BOUND       FEAS2700
CCCC                                                                    FEAS2705
C                                                                       FEAS2706
      IF( JLO .LE. 0 ) GO TO 2122                                       FEAS2710
C                                                                       FEAS2715
CCCC                                                                    FEAS2720
CCCCC       FOR VARIABLES AT THEIR LOWER BOUND, THE COLUMN IN THE       FEAS2730
CCCCC       A MATRIX IS OF THE FORM (0,DELFX,DELGIX, ... ,DELLIX, ...   FEAS2740
CCCCC       0, ... ,1, ... 0 ).  IN IAJ(I) KEEP A LIST OF THE VARIABLES FEAS2750
CCCC        IN THE ORDER THEY ARE PLACED IN THE "A" MATRIX              FEAS2760
CC                                                                      FEAS2770
      DO 2125 I=1,JLO                                                   FEAS2775
          N1 = IMIN(I)                                                  FEAS2780
          IAJ(I) = N1                                                   FEAS2790
          ACORE(1,I) =-DELFX(N1,X)                                      FEAS2800
C                                                                       FEAS2805
CC                    ARE THERE ANY BINDING NON-LINEAR CONSTRAINTS      FEAS2810
CC                                                                      FEAS2815
          IF( JNGI .LE. 0 ) GO TO 2125                                  FEAS2820
          DO 2120 N=1,JNGI                                              FEAS2830
              N2 = IGBIND(N)                                            FEAS2840
```

```
                XX = DELGIX(N1,N2,X)                                    FEAS2850
                IF( ABS(XX) .LT. 1.0E-39 ) XX = 0.0E0                   FEAS2860
                ACORE(N+1,I) = XX                                       FEAS2870
 2120       CONTINUE                                                    FEAS2880
 2125 CONTINUE                                                          FEAS2890
C                                                                       FEAS2895
CC                      ARE ANY VARIABLES AT THEIR UPPER BOUND          FEAS2900
C                                                                       FEAS2905
 2122 IF( JHI .LE. 0 ) GO TO 2132                                       FEAS2910
C                                                                       FEAS2920
CC          FOR VARIABLES AT THEIR UPPER BOUND THE COLUMN OF ACORE IS   FEAS2930
CC          OF THE FORM (-DELFX, -DELGIX, ... )                        FEAS2940
C                                                                       FEAS2950
      DO 2130 I=1,JHI                                                   FEAS2960
          N1 = IMAX(I)                                                  FEAS2970
          I1 = JLO + I                                                  FEAS2980
          IAJ(I1) = -N1                                                 FEAS2990
C                                                                       FEAS2995
CCC                       THE MINUS SIGN INDICATES THIS IS AN UPPER BOUND FEAS3000
C                                                                       FEAS3005
CCCCC       COLUMN AND WILL BE USED LATER WHEN GENERATING AJ.           FEAS3010
C                                                                       FEAS3020
          ACORE(1,I1) =  DELFX(N1,X)                                    FEAS3030
C                                                                       FEAS3035
CCC                       ARE THERE ANY BINDING NON-LINEAR CONSTRAINTS. FEAS3040
C                                                                       FEAS3045
          IF( JNGI .LE. 0 ) GO TO 2130                                  FEAS3050
          DO 2126 N=1,JNGI                                              FEAS3060
              N2 = IGBIND(N)                                            FEAS3070
              XX = -DELGIX(N1,N2,X)                                     FEAS3080
              IF( ABS(XX) .LT. 1.0E-39 ) XX = 0.0E0                     FEAS3090
              ACORE(N+1,I1 ) = XX                                       FEAS3100
 2126       CONTINUE                                                    FEAS3110
 2130 CONTINUE                                                          FEAS3120
C                                                                       FEAS3125
CC                      ARE THERE ANY VARIABLES NOT AT A BOUND.         FEAS3130
C                                                                       FEAS3135
 2132 IF( JIN .LE. 0 ) GO TO 2150                                       FEAS3140
C                                                                       FEAS3150
CC          FOR VARIABLES NOT AT A BOUND, WE GENERATE TWO COLUMNS,      FEAS3160
CCCC        ONE LIKE A LOWER BOUND COLUMN, AND ONE LIKE AN UPPER BOUND  FEAS3170
CC          COLUMN.                                                     FEAS3180
C                                                                       FEAS3190
      DO 2140 I=1,JIN                                                   FEAS3200
          N1 = IXIN(I)                                                  FEAS3210
          I1 = JLO + JHI + 2 * I                                        FEAS3220
          IAJ(I1 - 1) = N1                                              FEAS3230
          IAJ(I1) = -N1                                                 FEAS3240
```

```
            XX = DELFX(N1,X)                                                FEAS3250
            ACORE(1,I1-1) = -XX                                             FEAS3260
            ACORE(1,I1) =  XX                                               FEAS3270
C                                                                           FEAS3275
CC                  ARE THERE ANY BINDING NON-LINEAR CONSTRINTS             FEAS3280
C                                                                           FEAS3285
      IF(JNGI .LE. 0) GO TO 2140                                            FEAS3290
          DO 2136 N=1,JNGI                                                  FEAS3300
              N2 = IGBIND(N)                                                FEAS3310
              XX = DELGIX(N1,N2,X)                                          FEAS3320
              IF( ABS(XX) .LT. 1.0E-39 ) XX = 0.0                           FEAS3330
              ACORE(N+1,I1-1) = XX                                          FEAS3340
              ACORE(N+1,I1) = -XX                                           FEAS3350
 2136     CONTINUE                                                          FEAS3360
 2140 CONTINUE                                                              FEAS3370
C                                                                           FEAS3380
CC          CALCULATE ZJ - CJ                                               FEAS3390
CC                  THE ZJ - CJ VALUES FOR THE VARIABLES NOT IN             FEAS3400
CC          IN THE BASIS.  ZJ - ZJ IS EQUAL TO THE VECTOR PRODUCT OF        FEAS3410
CC          THE FIRST ROW OF THE BASIS INVERSE AND THE COLUMN VECTOR        FEAS3420
CC          FROM THE "A" MATRIX CORRESPONDING TO THE VARIABLE.  THE         FEAS3430
CC          ZJ - CJ VALUES ARE STORED IN ZJCJ(I)                            FEAS3440
CC                                                                          FEAS3450
CC                  ZERO OUT ZJCJ(I)                                        FEAS3460
C                                                                           FEAS3465
 2150 DO 2152 I=1,ILPCOL                                                    FEAS3470
          ZJCJ(I) = 0.0E0                                                   FEAS3480
 2152 CONTINUE                                                              FEAS3490
C                                                                           FEAS3495
CC                  IF 2 IS NOT IN BASIS CALCULATE ZJCJ(2)                  FEAS3500
C                                                                           FEAS3505
      IF( $BASIC(2) ) GO TO 2156                                            FEAS3510
      DZJ = -1.0D0                                                          FEAS3520
      IEND = JNGI + 2                                                       FEAS3530
      DO 2154 I=2,IEND                                                      FEAS3540
          DZJ = DZJ + BINVRS(1,I)                                           FEAS3550
 2154 CONTINUE                                                              FEAS3560
      ZJCJ(2) = SNGL(DZJ)                                                   FEAS3570
C                                                                           FEAS3575
CC                  NOW CALCULATE ZJ - CJ FOR THOSE NON-BASIC               FEAS3580
CC          VARIABLES IN THE "CORE".                                        FEAS3590
C                                                                           FEAS3595
 2156 DO 2166 I=3,ICCOL                                                     FEAS3600
C                                                                           FEAS3605
CC                  CALCULATE ZJ - CJ FOR NON-BASIC VARIBLES ONLY           FEAS3610
C                                                                           FEAS3615
          IF( $BASIC(I) ) GO TO 2166                                        FEAS3620
          IEND = JNGI + 1                                                   FEAS3630
```

```
            I2 = I - 2                                                      FEAS3640
            N1 = IAJ(I2)                                                    FEAS3650
            N2 = IABS(N1)                                                   FEAS3660
            IS = ISIGN(1,N1)                                                FEAS3670
            DZJ = 0.0D0                                                     FEAS3680
            DO 2158 N=1,IEND                                                FEAS3690
                DBI = BINVRS(1,N+1)                                         FEAS3700
                IF( DABS(DBI) .LT. 1.0D-39 ) GO TO 2158                     FEAS3710
                DEL = DBLE(ACORE(N,I2))                                     FEAS3720
                IF( DABS(DEL) .LT. 1.0D-39 ) GO TO 2158                     FEAS3730
                DZJ = DZJ + DEL * DBI                                       FEAS3740
 2158       CONTINUE                                                        FEAS3750
C                                                                           FEAS3755
CC                      IF THERE ARE ANY BINDING LINEAR CONSTRAINTS         FEAS3760
CC                      CALCULATE THEIR CONTRIBUTION                        FEAS3770
C                                                                           FEAS3775
            IF( JNLI .LE. 0 ) GO TO 2162                                    FEAS3780
            I1 = IEND + 1                                                   FEAS3790
            DO 2160 N=1,JNLI                                                FEAS3800
                J1 = ILBIND(N)                                              FEAS3810
                DBI = BINVRS(1,N+I1)                                        FEAS3820
                IF( DABS(DBI) .LT. 1.0D-39 ) GO TO 2160                     FEAS3830
                DEL = DBLE(GLCNST(J1,N2))                                   FEAS3840
                IF( DABS(DEL) .LT. 1.0D-39 ) GO TO 2160                     FEAS3850
                DZJ = DZJ + IS * DEL * DBI                                  FEAS3860
 2160       CONTINUE                                                        FEAS3870
C                                                                           FEAS3875
CC                      NOW CALCULATE THE CONTRIBUTION FROM THE S(J)        FEAS3880
CC                      CONSTRAINTS.                                        FEAS3890
C                                                                           FEAS3895
 2162       J1 = ICROW + N2                                                 FEAS3900
            DZJ = DZJ + BINVRS(1,J1)                                        FEAS3910
            ZJCJ(I) = SNGL(DZJ)                                             FEAS3920
 2166 CONTINUE                                                              FEAS3930
C                                                                           FEAS3935
CC                      NOW FOR THE NON-BASIC SLACK VARIABLES               FEAS3940
C                                                                           FEAS3945
      I1 = ICCOL + 1                                                        FEAS3950
      DO 2168 I=I1,ILPCOL                                                   FEAS3960
          IF( $BASIC(I) ) GO TO 2168                                        FEAS3970
          ZJCJ(I) = SNGL(BINVRS(1,I-I1+2))                                  FEAS3980
 2168 CONTINUE                                                              FEAS3990
C                                                                           FEAS4000
CC          SELECT COLUMN TO ENTER BASIS.                                   FEAS4010
CC                      PICK COLUMN WITH MOST NEGITIVE ZJ-CJ.  DO NOT       FEAS4020
CC          CONSIDER COLUMNS ALREADY IN THE BASIS, NOR THOSE WHOSE          FEAS4030
CC          YJ VECTOR HAVE NO POSITIVE YJ ELEMENTS.  IF ALL ZJ-CJ VALUES    FEAS4040
C           ARE GREATER THAN OR EQUAL TO ZERO, THEN WE HAVE FOUND THE       FEAS4050
```

```
C          OPTIMAL SOLUTION FOR THE DIRECTION FINDING PROBLEM.          FEAS4060
C                                                                       FEAS4070
 2170 DO 2172 I=1,ILPCOL                                                FEAS4080
         $YJPOS(I) = .TRUE.                                             FEAS4090
 2172 CONTINUE                                                          FEAS4100
 2174 ZCJMIN = ZCJTST                                                   FEAS4110
C                                                                       FEAS4115
C                    FIND THE MOST NEGATIVE ZJCJ(I)                     FEAS4120
C                                                                       FEAS4125
      DO 2176 I=2,ILPCOL                                                FEAS4130
         IF( $BASIC(I) ) GO TO 2176                                     FEAS4140
         IF( ZJCJ(I) .GE. ZCJMIN ) GO TO 2176                           FEAS4150
         IF( .NOT. $YJPOS(I) ) GO TO 2176                               FEAS4160
         ZCJMIN = ZJCJ(I)                                               FEAS4170
         IIN = I                                                        FEAS4180
C                                                                       FEAS4185
CC                   ENTERING COLUMN IS IIN                             FEAS4190
C                                                                       FEAS4195
 2176 CONTINUE                                                          FEAS4200
C                                                                       FEAS4205
CC                   WERE THERE ANY NEGATIVE ZJ-CJ VALUES.              FEAS4210
C                                                                       FEAS4215
      IF( ZCJMIN .GE. ZCJTST) GO TO 2350                                FEAS4220
C                                                                       FEAS4230
CC       CALCULATE AJ VECTOR                                            FEAS4240
CC                   COMPUTE THE ENTIRE AJ VECTOR FOR THE ENTERING      FEAS4250
CC       VARIABLE. CHECK WHICH COLUMN IS ENTERING.  IF 2 FORM OF        FEAS4260
CC       AJ IS (-1,1, ... ,1,0, ... ,0), IF BETWEEN 3 AND ICCOL         FEAS4270
CC       FROM IS (0,ACORE, ... , ALINEAR, ... ,0, ... ,1, ... ,0)      FEAS4280
CC       IF GREATER THAN ICCOL, (0,... ,0,1,0, ... ,0)                  FEAS4290
CC                                                                      FEAS4300
CC                   ZERO OUT THE AJ VECTOR                             FEAS4310
C                                                                       FEAS4315
 2180 DO 2182 I=1,ILPROW                                                FEAS4320
         AJ(I) = 0.0E0                                                  FEAS4330
 2182 CONTINUE                                                          FEAS4340
      IF( IIN .EQ. 2 ) GO TO 2192                                       FEAS4350
      IF( IIN .LE. ICCOL ) GO TO 2184                                   FEAS4360
C                                                                       FEAS4365
CC                   IIN GREATER THAN ICCOL                             FEAS4370
C                                                                       FEAS4375
      I1 = IIN - ICCOL + 1                                              FEAS4380
      GO TO 2220                                                        FEAS4390
C                                                                       FEAS4395
CC                   IIN BETWEEN THREE AND ICCOL                        FEAS4400
C                                                                       FEAS4405
 2184 I2 = IIN - 2                                                      FEAS4410
      N1 = IAJ(I2)                                                      FEAS4420
```

```
      IS = ISIGN(1,N1)                                                  FEAS4430
      N2 = IABS(N1)                                                     FEAS4440
      IEND = JNCI + 2                                                   FEAS4450
      DO 2186 I=2,IEND                                                  FEAS4460
          AJ(I) = ACORE(I-1,I2)                                         FEAS4470
 2186 CONTINUE                                                          FEAS4480
C                                                                       FEAS4485
CC                    ARE THERE ANY BINDING LINEAR CONSTRAINTS          FEAS4490
C                                                                       FEAS4495
      IF( JNLI .LE. 0 ) GO TO 2190                                      FEAS4500
      I1 = IEND                                                         FEAS4510
      DO 2188 I=1,JNLI                                                  FEAS4520
          J1 = ILBIND(I)                                                FEAS4530
          AJ(I+I1) = IS * GLCNST(J1,N2)                                 FEAS4540
 2188 CONTINUE                                                          FEAS4550
 2190 J1 = ICROW + N2                                                   FEAS4560
      AJ(J1) = 1.0E0                                                    FEAS4570
      GO TO 2200                                                        FEAS4580
C                                                                       FEAS4585
CC                    IIN IS TWO                                        FEAS4590
C                                                                       FEAS4595
 2192 AJ(1) = -1.0E0                                                    FEAS4600
      IEND = JNGI + 2                                                   FEAS4610
      DO 2194 I=2,IEND                                                  FEAS4620
          AJ(I) = 1.0E0                                                 FEAS4630
 2194 CONTINUE                                                          FEAS4640
C                                                                       FEAS4650
CC          CALCULATE JY VECTOR                                         FEAS4660
C                   COMPUTE YJ FOR ENTERING VECTOR. YJ IS THE VECTOR    FEAS4670
CC          PRODUCT OF B INVERSE AND AJ.  THEN SELECT THE VARIABLE      FEAS4680
CC          LEAVING THE BASIS.  LEAVING VARIABLE IS THE ONE WITH THE    FEAS4690
CC          MINIMUM VALUE OF XB(I)/YJ(I) WITH YJ(I) POSITIVE.           FEAS4700
C                                                                       FEAS4710
 2200 XLO = CMAX                                                        FEAS4720
      IOUT = -1                                                         FEAS4730
      DO 2208 I=1,ILPROW                                                FEAS4740
          DSUM = 0.0D0                                                  FEAS4750
          DO 2204 N=1,ILPROW                                            FEAS4760
              DBI = BINVRS(I,N)                                         FEAS4770
              IF( DABS(DBI) .LT. 1.0D-39) GO TO 2204                    FEAS4780
              DEL = DBLE(AJ(N))                                         FEAS4790
              IF( DABS(DEL) .LT. 1.0D-39 ) GO TO 2204                   FEAS4800
              DSUM = DSUM + DBI * DEL                                   FEAS4810
 2204     CONTINUE                                                      FEAS4820
          YJ(I) = SNGL(DSUM)                                            FEAS4830
          IF( DSUM .LE. 0.0D0 ) GO TO 2208                              FEAS4840
          XX = XB(I)                                                    FEAS4850
          IF( XX .EQ. 0.0E0 ) GO TO 2206                                FEAS4860
```

```
            IF( XX .LT. 0.0E0 ) GO TO 2208                                  FEAS4870
            XX = XX / DSUM                                                  FEAS4880
 2206       IF( XX .GE. XLO ) GO TO 2208                                    FEAS4890
            XLO = XX                                                        FEAS4900
            IOUT = I                                                        FEAS4910
 2208 CONTINUE                                                              FEAS4920
C                                                                           FEAS4925
CC                      NOW CHECK TO SEE IF THERE WERE ANY POSITIVE         FEAS4930
CC          YJ VALUES.  IF NOT GO BACK AND FIND ANOTHER ENTERING            FEAS4940
CC          VECTOR.                                                         FEAS4950
C                                                                           FEAS4955
 2210 IF( IOUT .GT. 0 ) GO TO 2300                                          FEAS4960
      $YJPOS(IIN) = .FALSE.                                                 FEAS4970
      GO TO 2174                                                            FEAS4980
C                                                                           FEAS4985
CC                      ENTERING VECTOR IS A UNIT VECTOR. IIN IS            FEAS4990
CC          GREATER THEN ICCOL.                                             FEAS5000
C                                                                           FEAS5005
 2220 IOUT = -1                                                             FEAS5010
      XLO = CMAX                                                            FEAS5020
      DO 2226 I=1,ILPROW                                                    FEAS5030
          DSUM = BINVRS(I,I1)                                               FEAS5040
          YJ(I) = SNGL(DSUM)                                                FEAS5050
          IF( DSUM .LE. 0.0D0 ) GO TO 2226                                  FEAS5060
          XX = XB(I)                                                        FEAS5070
          IF( XX .EQ. 0.0E0 ) GO TO 2222                                    FEAS5080
          IF( XX .LT. 0.0E0 ) GO TO 2226                                    FEAS5090
          XX = XX / DSUM                                                    FEAS5100
 2222     IF( XX .GE. XLO ) GO TO 2226                                      FEAS5110
          XLO = XX                                                          FEAS5120
          IOUT = I                                                          FEAS5130
 2226 CONTINUE                                                              FEAS5140
      GO TO 2210                                                            FEAS5150
C                                                                           FEAS5155
CC          CALCULATE PHI AND UPDATE XB                                     FEAS5160
CC                      THE VECTOR PHI IS USED TO GENERATE THE NEW          FEAS5170
CC          BASIS INVERSE AND XB.  PHI(I) IS -YJ(I) / PIVIOT ELEMENT        FEAS5180
CC          YJ(IOUT).  PHI(IOUT) IS ( 1 / YJ(IOUT) ) - 1.0.  PHI            FEAS5190
CC          WILL BE STORED IN SAME LOCATION AS YJ.  THEN UPDATE             FEAS5200
CC          XB.  NEW XB IS OLD XB + ( 1 / XB(IOUT) ) * PHI                  FEAS5210
C                                                                           FEAS5220
 2300 DBI = DBLE(YJ(IOUT))                                                  FEAS5230
 2301 DBI = 1.0D0 / DBI                                                     FEAS5240
      IF( DABS(DBI) .LT. 1.0D-39 ) DBI = DSIGN(1.0D-39,DBI)                 FEAS5250
      DO 2310 I=1,ILPROW                                                    FEAS5260
      IF( I .NE. IOUT ) GO TO 2304                                          FEAS5270
      PHI(I) = SNGL(DBI - 1.0D0)                                            FEAS5280
      GO TO 2310                                                            FEAS5290
```

```
 2304      DEL = DBLE(YJ(I))                                                  FEAS5300
           IF( DABS(DEL) .LT. 1.0D-39 ) GO TO 2308                            FEAS5310
           DEL = -DEL * DBI                                                   FEAS5320
           PHI(I) = SNGL(DEL)                                                 FEAS5330
           GO TO 2310                                                         FEAS5340
 2308      PHI(I) = 0.0E0                                                     FEAS5350
 2310 CONTINUE                                                                FEAS5360
      DBI = DBLE( XB(IOUT) )                                                  FEAS5370
      IF( DABS(DBI) .LT. 1.0D-39 ) GO TO 2320                                 FEAS5380
      DO 2312 I=1,ILPROW                                                      FEAS5390
          DEL = DBLE(PHI(I))                                                  FEAS5400
          IF( DABS(DEL) .LT. 1.0D-39 ) GO TO 2312                             FEAS5410
          DSUM = DBLE(XB(I))                                                  FEAS5420
          DSUM = DSUM + DBI * DEL                                             FEAS5430
          XB(I) = SNGL(DSUM)                                                  FEAS5440
 2312 CONTINUE                                                                FEAS5450
C                                                                             FEAS5460
C         CALCULATE NEW BASIS INVERSE                                         FEAS5470
C                   COMPUTE NEW BASIS INVERSE USIN PHI.  EACH COLUMN          FEAS5480
C         VECTOR IN THE NEW BASIS INVERSE IS EQUAL TO THE SUM OF THE          FEAS5490
C         OLD COLUMN WITH THE PRODUCT OF PHI AND THE COLUMN PIVIOT            FEAS5500
C         ELEMENT.                                                            FEAS5510
C                                                                             FEAS5515
 2320 DO 2328 N=1,ILPROW                                                      FEAS5520
          DBI = BINVRS(IOUT,N)                                                FEAS5530
          IF( DABS(DBI) .LT. 1.0D-39 ) GO TO 2328                             FEAS5540
          DO 2324 I=1,ILPROW                                                  FEAS5550
              DEL = DBLE(PHI(I))                                              FEAS5560
              IF( DABS(DEL) .LT. 1.0D-39 ) GO TO 2324                         FEAS5570
              BINVRS(I,N) = BINVRS(I,N) + DBI * DEL                           FEAS5580
 2324     CONTINUE                                                            FEAS5590
 2328 CONTINUE                                                                FEAS5600
C                                                                             FEAS5605
C                     NOW UPDATE THE BASIS LIST.                              FEAS5610
C                                                                             FEAS5615
      I1 = IBASIC(IOUT)                                                       FEAS5620
      $BASIC(I1) = .FALSE.                                                    FEAS5630
      $BASIC(IIN) = .TRUE.                                                    FEAS5640
      IBASIC(IOUT) = IIN                                                      FEAS5650
 2329 $INV = .FALSE.                                                          FEAS5660
C                                                                             FEAS5665
C                     THE INVERSE IS NOT A DIRECT INVERSION.                  FEAS5670
C         NOW GO BACK AND CALCULATE THE NEW ZJ-CJ VALUES.                     FEAS5680
C                                                                             FEAS5685
      GO TO 2150                                                              FEAS5690
C                                                                             FEAS5700
C         BASIS REINVERSION                                                   FEAS5710
C                   THE DIRECTION FINDING PROBLEM HAS REACHED AN              FEAS5720
```

```
C          OPTIMAL SOLUTION.  COMPUTE THE BASIS INVERSE DIRECTLY          FEAS5730
C          BY INVERTING CURRENT BASIS.  THE FIRST STEP IS TO BUILD        FEAS5740
C          THE CURRENT BASIS FROM THE "A" MATRIX AND THEN INVERTING       FEAS5750
C          IT DIRECTLY USING THE LIBRARY ROUTINE DMINV.  THE LIBRARY      FEAS5760
C          ROUTINE ARRAY HAS BEEN MODIFIED TO DOUBLE PRECISION AND        FEAS5770
C          IS TO PACK AND UNPACK BINVRS.  IF SOLUTION IS STILL            FEAS5780
C          OPTIMAL AFTER REINVERSION THEN GO ON.                          FEAS5790
C                                                                         FEAS5800
 2350 IF( $INV ) GO TO 2380                                               FEAS5810
      BINVRS(1,1) = 1.0D0                                                 FEAS5820
      DO 2352 I=2,ILPROW                                                  FEAS5830
          BINVRS(I,1) = 0.0D0                                             FEAS5840
 2352 CONTINUE                                                            FEAS5850
      DO 2370 I=2,ILPROW                                                  FEAS5860
          I1 = IBASIC(I)                                                  FEAS5870
          DO 2354 N=1,ILPROW                                              FEAS5880
              BINVRS(N,I) = 0.0D0                                         FEAS5890
 2354     CONTINUE                                                        FEAS5900
          IF( I1 .NE. 2 ) GO TO 2359                                      FEAS5910
          BINVRS(1,I) = -1.0D0                                            FEAS5920
          IEND = JNGI + 2                                                 FEAS5930
          DO 2358 N=2,IEND                                                FEAS5940
              BINVRS(N,I) = 1.0D0                                         FEAS5950
 2358     CONTINUE                                                        FEAS5960
          GO TO 2370                                                      FEAS5970
 2359     IF( I1 .LE. ICCOL ) GO TO 2360                                  FEAS5980
          N1 = I1 - ICCOL + 1                                             FEAS5990
          BINVRS(N1,I) = 1.0D0                                            FEAS6000
          GO TO 2370                                                      FEAS6010
 2360     I2 = I1 - 2                                                     FEAS6020
          N1 = IAJ(I2)                                                    FEAS6030
          IS = ISIGN(1,N1)                                                FEAS6040
          N2 = IABS(N1)                                                   FEAS6050
          IEND = JNGI + 2                                                 FEAS6060
          DO 2362 N=2,IEND                                                FEAS6070
              BINVRS(N,I) = DBLE(ACORE(N-1,I2))                           FEAS6080
 2362     CONTINUE                                                        FEAS6090
          IF( JNLI .EQ. 0 ) GO TO 2366                                    FEAS6100
          N1 = IEND                                                       FEAS6110
          DO 2364 N=1,JNLI                                                FEAS6120
              J1 = ILBIND(N)                                              FEAS6130
              BINVRS(N+N1,I) = IS * DBLE(GLCNST(J1,N2))                   FEAS6140
 2364     CONTINUE                                                        FEAS6150
 2366     J1 = ICROW + N2                                                 FEAS6160
          BINVRS(J1,I) = 1.0D0                                            FEAS6170
 2370 CONTINUE                                                            FEAS6180
 2371 N1 = ILPROW                                                         FEAS6190
      CALL DARRAY(2,N1,N1,152,152,BINVRS,BINVRS)                          FEAS6200
```

```
      CALL OMINV(BINVRS,N1,DEL,NWORKL,NWORKM)                          FEAS6210
      CALL DARRAY(1,N1,N1,152,152,BINVRS,BINVRS)                       FEAS6220
      $INV = .TRUE.                                                    FEAS6230
      GO TO 2150                                                       FEAS6240
C                                                                      FEAS6250
CC         CALCULATE THE DIRECTION                                     FEAS6260
CC                   THE DIRECTION S IS THE OPTIMAL SOLUTION TO        FEAS6270
CC         THE L. P. PROBLEM.  S(I) EQUALS THE VALUE OF THE            FEAS6280
CC         CORRESPONDING XB(J).  IF THE THE VARIABLE IS NOT IN THE     FEAS6290
CC         BASIS THEN S(I) EQUALS ZERO.  IF THE OTIMAL VALUE OF        FEAS6300
CC         THE DIRECTION FINDING OBJECTIVE FUNCTION (XDIR) IS LESS     FEAS6310
CC         THAN EPSCON, SET EPSLON EQUAL TO EPSLON 1 2 AND REDO        FEAS6320
CC         THE DIRECTION FINDING PROBLEM.  IF THE VALUE OF XDIR IS     FEAS6330
CC         ZERO (OR VERY NEAR ZERO) THERE IS NO USABLE FEASIBLE        FEAS6340
CC         DIRECTION, THUS WE HAVE REACHED THE OPTIMAL SOLUTION.       FEAS6350
C                                                                      FEAS6360
 2380 IF( XDIR .LE. 1.0E-10 ) GO TO 2650                               FEAS6370
      IF( XDIR .LT. EPSLON ) GO TO 2390                                FEAS6380
      DO 2382 I=1,IXMAX                                                FEAS6390
         S(I) = 0.0E0                                                  FEAS6400
 2382 CONTINUE                                                         FEAS6410
      DO 2384 I=2,ILPROW                                               FEAS6420
         I1 = IBASIC(I)                                                FEAS6430
         IF( I1 .GT. ICCOL ) GO TO 2384                                FEAS6440
         IF( I1 .LE. 2 ) GO TO 2384                                    FEAS6450
         N1 = IAJ(I1-2)                                                FEAS6460
         N2 = IABS(N1)                                                 FEAS6470
         IS = ISIGN(1,N1)                                              FEAS6480
         S(N2) = IS * XB(I)                                            FEAS6490
 2384 CONTINUE                                                         FEAS6500
      GO TO 2400                                                       FEAS6510
 2390 EPSLON = EPSLON / 2.0                                            FEAS6520
      GO TO 2000                                                       FEAS6530
C                                                                      FEAS6540
CC         DETERMINE THE STEP LENGTH                                   FEAS6550
CC                   THIS WILL BE DONE IN TWO STAGES.  FIRST THE       FEAS6560
CC         LONGEST STEP LENGTH SUCH THAT THE NEW POINT IS STILL        FEAS6570
CC         FEASIBLE IS DETERMINED (XLAMF).  THEN THE STEP LENGTH       FEAS6580
CC         LESS THAN OR EQUAL TO XLAMF THAT MAXIMIZES F(X) ALONG       FEAS6590
CC         THE FEASIBLE DIRECTION IS DETERMINED.  THIS IS DONE BY      FEAS6600
CC         FIND THE POINT THE DERRIVITIVE GOES TO ZERO.  IF THE        FEAS6610
CC         STEP LENGTH IS UNBOUNDED THEN THE PROBLEM IS UNBOUNDED.     FEAS6620
CC                                                                     FEAS6630
CC                   SAVE THE PRESENT POINT                            FEAS6640
C                                                                      FEAS6645
 2400 DO 2402 I=1,IXMAX                                                FEAS6650
         XSAVE(I) = X(I)                                               FEAS6660
 2402 CONTINUE                                                         FEAS6670
```

```
C                                                                       FEAS6671
CC                    SET INITIAL VALUES, $FEAS IS TRUE, LARGEST KNOWN  FEAS6672
CC                    FEASIBLE STEP IS ZERO, FIRST STEP IS ONE.         FEAS6673
C                                                                       FEAS6674
      $FEAS = .TRUE.                                                    FEAS6680
      XLAMDA = 1.0E0                                                    FEAS6690
      XLAMF = 0.0E0                                                     FEAS6700
C                                                                       FEAS6705
CC                    MOVE TO THE NEW POINT.                            FEAS6706
C                                                                       FEAS6707
 2404 DO 2408 I= 1,IXMAX                                                FEAS6710
          XX = S(I)                                                     FEAS6720
          IF( ABS(XX) .LT. 1.0E-39 ) GO TO 2406                         FEAS6730
          X(I) = XSAVE(I) + XX * XLAMDA                                 FEAS6740
C                                                                       FEAS6745
CC                    IS THE NEW POINT UNBOUNDED.                       FEAS6746
C                                                                       FEAS6747
          IF( X(I) .GT. CMAX / 2.0 ) GO TO 2452                         FEAS6750
          GO TO 2408                                                    FEAS6760
 2406     X(I) = XSAVE(I)                                               FEAS6770
 2408 CONTINUE                                                          FEAS6780
C                                                                       FEAS6785
CC                    NOW CHECK FEASIBILITY OF THE NEW POINT.           FEAS6786
CC                    FIRST CHECK BOUNDS ON  X(I).                      FEAS6787
C                                                                       FEAS6788
      DO 2412 I=1,IXMAX                                                 FEAS6790
          XX = X(I)                                                     FEAS6800
          XLO = XMIN(I)                                                 FEAS6810
          XHI = XMAX(I)                                                 FEAS6820
          IF( XX .LT. XLO ) GO TO 2430                                  FEAS6830
          IF( XX .GT. XHI ) GO TO 2430                                  FEAS6840
 2412 CONTINUE                                                          FEAS6850
C                                                                       FEAS6855
CC                    IF THERE ANY NON-LINEAR CONSTRAINTS, CHECK        FEAS6856
CC                    THEM FOR FEASIBILITY                              FEAS6857
C                                                                       FEAS6858
      IF( IGCNT .LE. 0 ) GO TO 2418                                     FEAS6860
      DO 2416 N=1,IGCNT                                                 FEAS6870
          XX = GIX(N,X)                                                 FEAS6880
          XHI = BC(N)                                                   FEAS6890
          IF( XX .GT. XHI ) GO TO 2430                                  FEAS6900
 2416 CONTINUE                                                          FEAS6910
C                                                                       FEAS6915
CC                    IF THERE ARE LINEAR CONSTRAINTS, CHECK THEM.      FEAS6916
C                                                                       FEAS6917
 2418 IF( ILCNT .LE. 0 ) GO TO 2429                                     FEAS6920
 2420 DO 2428 N=1,ILCNT                                                 FEAS6930
          DSUM = 0.0D0                                                  FEAS6940
```

```
            DO 2422 I=1,IXMAX                                                   FEAS6950
                DBI = DBLE(X(I))                                                FEAS6960
                IF( DABS(DBI) .LT. 1.0D-39 ) GO TO 2422                         FEAS6970
                DEL = DBLE(GLCNST(N,I))                                         FEAS6980
                IF( DABS(DEL) .LT. 1.0D-39 ) GO TO 2422                         FEAS6990
                DSUM = DSUM + DEL * DBI                                         FEAS7000
 2422       CONTINUE                                                            FEAS7010
            XX = SNGL(DSUM)                                                     FEAS7020
            XHI = BL(N)                                                         FEAS7030
            IF( XX .GT. XHI ) GO TO 2430                                        FEAS7040
 2428   CONTINUE                                                                FEAS7050
C                                                                               FEAS7051
CC                      THE NEW POINT IS FEASIBLE.  IF IT HAS A LARGER          FEAS7052
CC                      VALUE FOT THE OBJECTIVE FUNCTION, TAKE ANOTHER          FEAS7053
C                       STEP, OTHERWISE GO ON TO STAGE TWO.                     FEAS7054
C                                                                               FEAS7055
 2429   IF( FX(X) .GT. FOFX ) GO TO 2440                                        FEAS7056
        XLAMF = XLAMDA                                                          FEAS7060
        GO TO 2450                                                              FEAS7065
C                                                                               FEAS7069
CC                      IF WE GET HERE POINT IS NOT FEASIBLE.                   FEAS7070
C                                                                               FEAS7071
 2430   XLAMNF = XLAMDA                                                         FEAS7080
        XLAMDA = ( XLAMNF + XLAMF ) / 2.0                                       FEAS7090
        $FEAS = .FALSE.                                                         FEAS7100
        IF( XLAMDA .LT. STPMIN ) GO TO 2700                                     FEAS7110
        IF( ( XLAMNF - XLAMF ) .LT. STPMIN ) GO TO 2450                         FEAS7112
C                                                                               FEAS7115
CC                      NOW TRY A SMALLER XLAMDA.                               FEAS7120
C                                                                               FEAS7125
        GO TO 2404                                                              FEAS7130
C                                                                               FEAS7135
CC                      POINT IS FEASIBLE, WAS PREVIOUS POINT FEASIBLE          FEAS7140
C                                                                               FEAS7145
 2440   IF( $FEAS ) GO TO 2444                                                  FEAS7150
        XINC = XLAMNF - XLAMF                                                   FEAS7160
        IF( XINC .LT. STPMIN ) GO TO 2450                                       FEAS7170
        XLAMF = XLAMDA                                                          FEAS7180
        XLAMDA = ( XLAMNF + XLAMF ) / 2.0                                       FEAS7190
        GO TO 2404                                                              FEAS7200
 2444   IF( XLAMDA .GE. CMAX / 2.0 ) GO TO 2448                                 FEAS7210
        XLAMF = XLAMDA                                                          FEAS7220
        XLAMDA = XLAMDA * 2.0                                                   FEAS7230
        GO TO 2404                                                              FEAS7240
 2448   XLAMF = XLAMDA                                                          FEAS7250
        GO TO 2452                                                              FEAS7260
C                                                                               FEAS7265
C                       NOW FIND THE STEP LENGTH LESS THAN OR EQUAL             FEAS7270
```

```
C          TO XLAMF THAT MAXIMIZE F(X) ALONG THE DIRECTION S.                FEAS7280
C                                                                            FEAS7285
 2450 $BOUND = .TRUE.                                                        FEAS7290
      XLAMDA = XLAMF                                                         FEAS7300
      GO TO 2454                                                             FEAS7310
C                                                                            FEAS7315
CC                    IF WE GET HERE, FIRST STAGE STEP LENGTH WAS            FEAS7316
CC                    UNBOUNDED.                                             FEAS7317
C                                                                            FEAS7318
 2452 $BOUND = .FALSE.                                                       FEAS7320
C                                                                            FEAS7322
CC                    XONE WILL CONTAIN THE LOWER BOUND OF THE STEP          FEAS7323
CC                    LENGTH, SLOPE OF F(X) IS POSITIBE AT XONE.  XTWO       FEAS7224
CC                    WILL BE UPPER BOUND, SLOPE NEGITIVE, WE WANT TO        FEAS7325
CC                    FIND THE POINT WITH ZERO SLOPE, WHICH WILL BE          FEAS7326
CC                    THE MAX OF F(X) ALONG THE DIRECTION S.                 FEAS7327
C                                                                            FEAS7328
 2454 DO 2460 I=1,IXMAX                                                      FEAS7330
          XX = S(I)                                                          FEAS7340
          XONE(I) = XSAVE(I)                                                 FEAS7350
          IF( ABS(XX) .LT. 1.0E-39 ) GO TO 2458                              FEAS7360
          X(I) = XSAVE(I) + XX * XLAMDA                                      FEAS7370
          GO TO 2460                                                         FEAS7380
 2458     X(I) = XSAVE(I)                                                    FEAS7390
 2460 CONTINUE                                                               FEAS7400
      DSUM = 0.0D0                                                           FEAS7410
      XLAMNF = 0.0E0                                                         FEAS7420
      DO 2464 N=1,IXMAX                                                      FEAS7430
          DEL = DBLE(S(N))                                                   FEAS7440
          IF( DABS(DEL) .LT. 1.0D-39 ) GO TO 2464                            FEAS7450
          DBI = DBLE(DELFX(N,X))                                             FEAS7460
          IF( DABS(DBI) .LT. 1.0D-39 ) GO TO 2464                            FEAS7470
          DSUM = DSUM + DEL * DBI                                            FEAS7480
 2464 CONTINUE                                                               FEAS7490
      IF( DSUM .GT. 0.0D0 ) GO TO 2476                                       FEAS7500
      BB = SNGL(DSUM)                                                        FEAS7510
      DSUM = 0.0D0                                                           FEAS7520
      XLAMUP = XLAMDA                                                        FEAS7530
      DO 2468 N=1,IXMAX                                                      FEAS7540
          DBI = DBLE(S(N))                                                   FEAS7550
          IF( DABS(DBI) .LT. 1.0D-39 ) GO TO 2468                            FEAS7560
          DEL = DBLE(DELFX(N,XONE))                                          FEAS7570
          IF( DABS(DEL) .LT. 1.0D-39 ) GO TO 2468                            FEAS7580
          DSUM = DSUM + DEL * DBI                                            FEAS7590
 2468 CONTINUE                                                               FEAS7600
      AA = SNGL(DSUM)                                                        FEAS7610
C                                                                            FEAS7612
C                     THE NEXT STEP LENGTH TO TRY IS THAT STEP WHICH         FEAS7613
```

```
C                   WOULD HAVE A ZERO SLOPE IF THE SLOPE WAS A LINEAR    FEAS7614
CC                  FUNCTION  OF STEP LENGTH.                            FEAS7615
C                                                                        FEAS7616
 2470 XLAMDA = ( BB * XLAMNF - AA * XLAMUP ) / ( BB - AA )               FEAS7620
C                                                                        FEAS7626
CC                  WE HAVE A NEW XTWO, UPPER BOUND.                     FEAS7627
C                                                                        FEAS7628
      DO 2474 I=1,IXMAX                                                  FEAS7630
          XX = S(I)                                                      FEAS7640
          XTWO(I) = X(I)                                                 FEAS7650
          IF( ABS(XX) .LT. 1.0E-39 ) GO TO 2472                          FEAS7660
          X(I) = XSAVE(I) + XX * XLAMDA                                  FEAS7670
          GO TO 2474                                                     FEAS7680
 2472     X(I) = XSAVE(I)                                                FEAS7690
 2474 CONTINUE                                                           FEAS7700
      GO TO 2480                                                         FEAS7710
C                                                                        FEAS7715
CC                  IF WE GET HERE F(X) IS INCREASING, BUT WE ARE        FEAS7716
CC                  AT A CONSTRAINT, IF PROB IS STILL BOUNDED GO ON      FEAS7727
CC                  TO NEXT ITERATION, OTHERWISE WE ARE DONE.            FEAS7718
C                                                                        FEAS7719
 2476 IF( FX(X) .GT. FX(XSAVE) ) GO TO 2478                              FEAS7720
      XLAMDA = XLAMDA / 2.0                                              FEAS7730
      GO TO 2454                                                         FEAS7740
 2478 IF( $BOUND ) GO TO 2500                                            FEAS7750
      GO TO 2800                                                         FEAS7760
 2480 DSUM = 0.0D0                                                       FEAS7770
      DO 2482 N=1,IXMAX                                                  FEAS7780
          DBI = DBLE(S(N))                                               FEAS7790
          IF( DABS(DBI) .LT. 1.0D-39 ) GO TO 2482                        FEAS7800
          DEL = DBLE(DELFX(N,X))                                         FEAS7810
          IF( DABS(DEL) .LT. 1.0D-39 ) GO TO 2482                        FEAS7820
          DSUM = DSUM + DEL * DBI                                        FEAS7830
 2482 CONTINUE                                                           FEAS7840
      IF( DSUM .EQ. 0.0D0 ) GO TO 2500                                   FEAS7850
      IF( DSUM .LT. 0.0D0 ) GO TO 2487                                   FEAS7860
C                                                                        FEAS7866
CC                  IF NEXT STEP IS TOO SMALL WE ARE DONE.               FEAS7867
C                                                                        FEAS7868
      IF( (XLAMUP - XLAMDA) .LT. STPMIN ) GO TO 2490                     FEAS7870
      AA = SNGL(DSUM)                                                    FEAS7880
      XLAMNF = XLAMDA                                                    FEAS7890
      XLAMDA = ( BB *  XLAMNF - AA * XLAMUP ) / ( BB - AA )              FEAS7900
C                                                                        FEAS7906
CC                  WE HAVE A NEW XONE, LOWER BOUND.                     FEAS7907
C                                                                        FEAS7908
      DO 2486 I=1,IXMAX                                                  FEAS7910
          XX = S(I)                                                      FEAS7920
```

```
            XONE(I) = X(I)                                                      FEAS7930
            IF( ABS(XX) .LT. 1.0E-39 ) GO TO 2485                               FEAS7940
            X(I) = XSAVE(I) + XX * XLAMDA                                       FEAS7950
            GO TO 2486                                                          FEAS7960
 2485       X(I) = XSAVE(I)                                                     FEAS7970
 2486  CONTINUE                                                                 FEAS7980
       GO TO 2480                                                               FEAS7990
C                                                                               FEAS7996
CC                    IF NEXT STEP IS TOO SMALL WE ARE DONE.                    FEAS7997
C                                                                               FEAS7998
 2487  IF( ( XLAMDA - XLAMNF ) .LT. STPMIN ) GO TO 2488                         FEAS8000
       BB = SNGL(DSUM)                                                          FEAS8010
       XLAMUP = XLAMDA                                                          FEAS8020
       GO TO 2470                                                               FEAS8030
 2488  IF( FX(X) .GT. FX(XONE) ) GO TO 2500                                     FEAS8040
       DO 2489 I=1,IXMAX                                                        FEAS8050
            X(I) = XONE(I)                                                      FEAS8060
 2489  CONTINUE                                                                 FEAS8070
       GO TO 2500                                                               FEAS8080
 2490  IF( FX(X) .GT. FX(XTWO) ) GO TO 2500                                     FEAS8090
       DO 2492 I=1,IXMAX                                                        FEAS8100
            X(I) = XTWO(I)                                                      FEAS8110
 2492  CONTINUE                                                                 FEAS8120
C                                                                               FEAS8125
CC                    WE HAVE A NEW POINT, STORED IN X.  GET SET                FEAS8130
CC         FOR NEXT ITERATION                                                   FEAS8140
C                                                                               FEAS8145
 2500  XX = FX(X)                                                               FEAS8150
       AA = ABS(FOFX)                                                           FEAS8156
       IF( AA .EQ. 0.0E0 ) AA = 1.0E-02                                         FEAS8158
       AX = ( ( XX - FOFX )/ AA ) * 1.0E02                                      FEAS8160
       FOFX = XX                                                                FEAS8170
       IF( AX .LT. PERCNT ) GO TO 2600                                          FEAS8180
       IF ( ITER .GT. ITMAX ) GO TO 2550                                        FEAS8190
       WRITE(NWRITE,2900) ITER, FOFX, AX                                        FEAS8200
 2900  FORMAT('0 ITERATION NUMBER:',I6,/,6X,'F(X) = ',1PE15.6,';  IMPROVEFEAS8210
     1MENT THIS ITERATION:',0PF10.4,' PERCENT.')                                FEAS8220
       ITER = ITER + 1                                                          FEAS8230
       WRITE(NWRITE,2902) ( I, X(I), I=1,IXMAX )                                FEAS8235
       GO TO 2000                                                               FEAS8240
 2550  WRITE(NWRITE,2901) ITER, FOFX                                            FEAS8250
 2901  FORMAT('0 PROGRAM TERMINATED AFTER',I6,' ITERATIONS.',/'0 THE PRESFEAS8260
     1ENT VALUE OF THE OBJECTIVE FUNCTION IS:',1PE15.6)                         FEAS8270
 2555  WRITE(NWRITE,2902) ( I, X(I), I=1,IXMAX )                                FEAS8280
 2902  FORMAT('0',5X,'THE POINT X( ) IS:',/,('0',5X,4('X(',I2,'), ',G15.7FEAS8290
     1,'; ')/))                                                                 FEAS8295
       WRITE(NWRITE,905)                                                        FEAS8298
  905  FORMAT('1')                                                              FEAS8299
```

```
      STOP                                                                  FEAS8300
2600  WRITE(NWRITE,2903) FOFX, AX, PERCNT                                   FEAS8310
2903  FORMAT('0 PROGRAM TERMINATED BECAUSE OF INSUFFICIENT IMPROVEMENT IFEAS8320
     1N OBJECTIVE FUNCTION.',/,'0 VALUE OF OBJECTIVE FUNCTION:',1PE20.7,FEAS8330
     2/'0 IMPROVEMENT LAST ITERATION WAS',0PF10.4,' PERCENT, WHICH IS LEFEAS8340
     3SS THAN THE LIMIT ',F10.4)                                           FEAS8350
2610  WRITE(NWRITE,2902) ( I, X(I), I=1,IXMAX )                             FEAS8360
      WRITE(NWRITE,905)                                                     FEAS8365
      STOP                                                                  FEAS8370
2650  WRITE(NWRITE,2907) FOFX                                               FEAS8372
2907  FORMAT('0 OPTIMAL SOLUTION:',/,'0 THE OPTIMAL VALUE OF THE OBJECTIFEAS8373
     1VE FUNCTION IS:',1PE15.6)                                            FEAS8374
      WRITE(NWRITE,2902) ( I, X(I), I=1,IXMAX)                              FEAS8375
      WRITE(NWRITE,905)                                                     FEAS8376
      STOP                                                                  FEAS8377
2700  WRITE(NWRITE,2904) FOFX                                               FEAS8380
2904  FORMAT('0 PROGRAM TERMINATED.  LENGTH OF LAST STEP WAS LESS THAN TFEAS8390
     1HE MINIMUM STEP.',/,'0 PRESENT VALUE OF THE OBJECTIVE FUNCTION:', FEAS8391
     21PE15.6)                                                             FEAS8392
      GO TO 2610                                                            FEAS8400
2800  WRITE(NWRITE,2905)                                                    FEAS8410
2905  FORMAT('0 PROBLEM IS UNBOUNDED.  LAST POINT AND DIRECTION ARE:')      FEAS8420
      WRITE(NWRITE,2902) ( I, XSAVE(I), I=1,IXMAX)                          FEAS8430
      WRITE(NWRITE,2906) ( I, S(I), I=1,IXMAX)                              FEAS8440
2906  FORMAT('0',5X,'LAST DIRECTION:',/,('0',5X,4('S(',I2,') ',G20.6,'  FEAS8450
     1,')/))                                                               FEAS8460
      WRITE(NWRITE,905)                                                     FEAS8465
      STOP                                                                  FEAS8470
      END                                                                   FEAS9000
```

```
C                                                                       CHE00010
C     ****************************************************************CHE00020
C                                                                       CHE00030
C                    PROGRAM CHECK                                      CHE00040
C                                                                       CHE00050
C     PURPOSE:                                                          CHE00060
C                                                                       CHE00070
C      TO CHECK OUT THE REQUIRED FUNCTION ROUTINES AND                  CHE00080
C      INPUT DATA FOR PROGRAM FEASBL.  THIS PROGRAM WILL                CHE00090
C      CHECK THE INPUT DATA FOR EPRORS TO ALLOW THE USER                CHE00100
C      OF FEASBL TO CHECK HIS INPUT FOR ERRORS QUICKLY                  CHE00110
C      WITHOUT HAVING TO USE THE FULL PROGRAM AND RESULTANT             CHE00120
C      SLOW TURN ARROUND ASSOCIATED WITH CLASS J JOBS.                  CHE00130
C                                                                       CHE00140
C     INPUT:                                                            CHE00150
C                                                                       CHE00160
C      THE INPUT REQUIRED IS THE SAME AS FOR THE PROGRAM                CHE00170
C      FEASBLE.                                                         CHE00180
C                                                                       CHE00190
C     ****************************************************************CHE00200
C                                                                       CHE00210
      IMPLICIT REAL*4 (A-H,O-Z), INTEGER*2 (I-L), INTEGER*4 (M-N),      CHE00220
     1LOGICAL*1 ($)                                                     CHE00230
      REAL*8 DBI,DEL, DSUM                                              CHE00240
      DIMENSION B(100), BC(50), BL(50)                                  CHE00250
      DIMENSION GLCNST(50,50), X(50), XMAX(50), XMIN(50)                CHE00260
      DIMENSION GL(50,50)                                               CHE00270
      EQUIVALENCE    (B(1), BC(1) ), ( B(51), BL(1) )                   CHE00280
      EQUIVALENCE ( GL(1,1), GLCNST(1,1) )                              CHE00290
      NAMELIST / INPUT / IXMAX, IGCNT, ILCNT, GL, XMIN, XMAX,           CHE00300
     1 BC, BL, X, EPSLON, STPMIN, PERCNT, ITMAX                         CHE00310
C                                                                       CHE00320
C         SET THE INITIAL VALUES                                        CHE00330
C                                                                       CHE00340
      DATA B / 100 * 0.0E0 /                                            CHE00350
      DATA XMIN / 50*0.0E0 /, XMAX / 50*Z7FFFFFFF /                     CHE00360
      DATA EPSLON / 1.0E-04 /,STPMIN / 1.0E-08 /, PERCNT / 1.0E-02 /    CHE00370
      DATA NREAD / 5 /, NWRITE / 6 /                                    CHE00380
      DATA GLCNST / 2500*0.0E0 /                                        CHE00390
      DATA X / 50*0.0E0 /                                               CHE00400
C                                                                       CHE00410
C     INPUT SECTION:                                                    CHE00420
C         READ IN FOLLOWING VALUES: IXMAX, IGCNT, ILCNT, GLCNST(I,J),   CHE00430
C         XMIN(I), XMAX(I), BC(I), BL(I), X(I), EPSLON, STPMIN, PERCNT  CHE00440
C         PRINT OUT ALL INPUT VALUES AND INITIAL VALUE OF OBJECT2VE     CHE00450
C         FUNCTION.                                                     CHE00460
C                                                                       CHE00470
      READ(NREAD,INPUT)                                                 CHE00480
```

```
      WRITE(NWRITE,901) IXMAX, IGCNT, ILCNT                                 CHE00490
  901 FORMAT('1',50X,'NON-LINEAR PROGRAMMING PROBLEM',///,'0  INITIAL SECHE00500
     1T UP:',/,'0',5X,'NUMBER OF VARIABLES:',13X,I5,/,'0',5X,'NUMBER OF CHE00510
     2NON-LINEAR CONSTRAINTS:',I5,/,'0',5X,'NUMBER OF LINEAR CONSTRAINTSCHE00520
     3:',4X,I5)                                                             CHE00530
C                                                                           CHE00540
C                    VERIFY THAT THE PARAMETERS ARE WITHIN LIMITS           CHE00550
C                                                                           CHE00560
      IF ( IXMAX .LE. 50 .AND. IGCNT .LE. 50 .AND. ILCNT .LE. 50 ) GO TOCHE00570
     1 1000                                                                 CHE00580
      WRITE(NWRITE,902)                                                     CHE00590
  902 FORMAT('0 * * * * * * * ERROR IN ABOVE PARAMETERS, PROGRAM TERMINATECHE00600
     1D * * * * * *')                                                       CHE00610
      STOP                                                                  CHE00620
 1000 IF( ILCNT .EQ. 0 ) GO TO 1005                                         CHE00630
      WRITE(NWRITE,904)                                                     CHE00640
  904 FORMAT('0',5X,'THE LINEAR CONSTRAINT MATRIX IS:')                     CHE00650
      DO 1004 I=1,ILCNT                                                     CHE00660
          DO 1003 N=1,IXMAX                                                 CHE00670
          IF( MOD(N,3) .EQ. 1 ) WRITE(NWRITE,1902) I, N, GL(I,N)            CHE00680
          IF( MOD(N,3) .EQ. 2 ) WRITE(NWRITE,1903) I, N, GL(I,N)            CHE00690
          IF( MOD(N,3) .EQ. 0 ) WRITE(NWRITE,1904) I, N, GL(I,N)            CHE00700
 1902     FORMAT('0',05X,'(',I2,',',I2,');',G15.6)                          CHE00710
 1903     FORMAT('+',33X,'(',I2,',',I2,');',G15.6)                          CHE00720
 1904     FORMAT('+',61X,'(',I2,',',I2,');',G15.6)                          CHE00730
 1003     CONTINUE                                                          CHE00740
          WRITE(NWRITE,1901)                                                CHE00750
 1901     FORMAT('0')                                                       CHE00760
 1004 CONTINUE                                                              CHE00770
 1909 FORMAT('0',5X,'THE BOUNDS ON THE VARIABLES ARE:',//)                  CHE00780
 1005 WRITE(NWRITE,1909)                                                    CHE00790
      DO 1018 N=1,IXMAX                                                     CHE00800
          IF( MOD(N,2) .EQ. 1 ) WRITE(NWRITE,1912) XMIN(N), N, XMAX(N)      CHE00810
 1912     FORMAT(6X,G15.6,' < X(',I2,') < ',G15.6,';')                      CHE00820
          IF( MOD(N,2) .EQ. 0 ) WRITE(NWRITE,1913) XMIN(N), N, XMAX(N)      CHE00830
 1913     FORMAT('+',55X,G15.6,' < X(',I2,') < ',G15.6)                     CHE00840
 1018 CONTINUE                                                              CHE00850
C                                                                           CHE00860
C                    CHECK THE UPPER AND LOWER BOUNDS FOR OBVIOUS           CHE00870
C                    ERRORS.  STOP IF ANY EXIST.                            CHE00880
C                                                                           CHE00890
      DO 1019 I=1,IXMAX                                                     CHE00900
          IF(XMIN(I) .LT. XMAX(I) ) GO TO 1019                              CHE00910
 1919 FORMAT('0 * * * * * ERROR IN BOUNDS, PROGRAM TERMINATED. * * * *')    CHE00920
      STOP                                                                  CHE00930
 1019 CONTINUE                                                              CHE00940
 1020 IF( IGCNT .EQ. 0 ) GO TO 1030                                         CHE00950
      WRITE(NWRITE,1921) (I,BC(I), I=1,IGCNT)                               CHE00960
```

```
 1921 FORMAT('0',5X,'NON-LINEAR CONSTRAINTS REQUIREMENTS VECTOR, BC(I):'CHE00970
     1,/,('0',5X,4('(',I2,');  ',G20.6)/))                              CHE00980
 1030 IF( ILCNT .EQ. 0 ) GO TO 1035                                     CHE00990
      WRITE(NWRITE,1930) (I, BL(I), I=1, ILCNT)                         CHE01000
 1930 FORMAT('0',5X,'LINEAR CONSTRAINTS REQUIREMENTS VECTOR, BL(I):',/, CHE01010
     1('0',5X,4('(',I2,');  ',G20.6)/))                                 CHE01020
 1035 WRITE(NWRITE,1935) ( I, X(I), I=1, IXMAX )                        CHE01030
 1935 FORMAT('0',5X,'THE INITIAL FEASIBLE STARTING POINT IS:',/,        CHE01040
     1('0',10X,4('X(',I2,'); ',G20.6,',')/))                            CHE01050
      WRITE(NWRITE,1937) EPSLON, STPMIN, PERCNT, ITMAX                  CHE01060
 1937 FORMAT('0',5X,'ANTI JAMMING CONSTANT:  ',G20.6,/,'0',5X,'MINIMUM  CHE01070
     1STEP  LENGTH:  ',G20.6,/,'0',5X,'MINIMUM   IMPROVEMENT:  ',G20.6, CHE01080
     2/,'0',5X,'MAXIMUM  # ITERATIONS:  ',G20.6)                        CHE01090
      FOFX = FX(X)                                                      CHE01100
      WRITE(NWRITE,1938) FOFX                                           CHE01110
 1938 FORMAT('0 INPUT COMPLETE.  INITIAL VALUE OF OBJECTIVE FUNCTION:', CHE01120
     11PE15.6,/'0')                                                     CHE01130
C                                                                       CHE01140
CC         DETERMINE THE BINDING CONSTRAINTS.                           CHE01150
CC         CHECK THE UPPER AND LOWER BOUNDS, NON-LINEAR, AND LINEAR     CHE01160
C     CONSTPAINTS.  PRINT OUT THE CONSTRAINTS THAT X IS                 CHE01170
C     OUTSIDE OF, LIST THE BOUNDS X EXCEEDS.                            CHE01180
CC                                                                      CHE01190
CC                                                                      CHE01200
CC         CHECK THE UPPER AND LOWER BOUNDS ON X.                       CHE01210
C                                                                       CHE01220
      DO 2010 I=1,IXMAX                                                 CHE01230
          XX = X(I)                                                     CHE01240
          XLO = XMIN(I)                                                 CHE01250
          XHI = XMAX(I)                                                 CHE01260
C                                                                       CHE01270
CC                    IS X(I) ABOVE ITS UPPER BOUND                     CHE01280
C                                                                       CHE01290
          IF( XX .GT. XHI ) GO TO 2090                                  CHE01300
         GO TO 2002                                                     CHE01310
C                                                                       CHE01320
CC        X(I) IS OUTSIDE BOUNDS, WRITE ERROR MESSAGE.                  CHE01330
C                                                                       CHE01340
 2090     WRITE(NWRITE,2990) I, XX                                      CHE01350
 2990 FORMAT('0 * * * * * X(',I2,') = ',1PE20.7,', IS OUTSIDE ITS BOUNDSCHE01360
     1, POINT IS NOT FEASIBLE. * * * * *')                              CHE01370
          GO TO 2010                                                    CHE01380
C                                                                       CHE01390
CC                    IS X(I) BELOW LOWER BOUND                         CHE01400
C                                                                       CHE01410
 2002     IF(XX .LT. XLO ) GO TO 2090                                   CHE01420
 2010 CONTINUE                                                          CHE01430
C                                                                       CHE01440
```

```
C                   ARE THERE ANY NON-LINEAR CONSTRAINTS                CHE01450
C                                                                       CHE01460
      IF( IGCNT .LE. 0 ) GO TO 2030                                     CHE01470
      DO 2020 N=1,IGCNT                                                 CHE01480
          XX = GIX(N,X)                                                 CHE01490
          XHI = BC(N)                                                   CHE01500
C                                                                       CHE01510
CC                  IS X INFEASIBLE WITH RESPECT TO CONSTRINT N        CHE01520
C                                                                       CHE01530
          IF( XX .GT. XHI ) GO TO 2092                                  CHE01540
          WRITE(NWRITE,2091)  N, XX                                     CHE01550
 2091     FORMAT('0 VALUE OF NON-LINEAR CONSTRAINT NUMBER,',I3,', IS:',  CHE01560
     1    1PE15.7)                                                      CHE01561
          GO TO 2020                                                    CHE01570
 2092     WRITE(NWRITE,2992) N, XX                                      CHE01580
 2992     FORMAT('0 * * * * * X IS NOT FEASIBLE WITH RESPECT TO NON-LINECHE01590
     1AR CONSTRAINT NUMBER',I3,' VALUE OF CONSTRAINT IS:',1PE20.7)      CHE01600
 2020 CONTINUE                                                          CHE01610
C                                                                       CHE01620
CC                  ARE THERE ANY LINEAR CONSTRAINTS                   CHE01630
C                                                                       CHE01640
 2030 IF( ILCNT .LE. 0 ) GO TO 2042                                     CHE01650
C                                                                       CHE01660
CC                 CHECK THE LINEAR CONSTRAINTS.                       CHE01670
C                                                                       CHE01680
      DO 2040 N=1,ILCNT                                                 CHE01690
          DSUM = 0.0D0                                                  CHE01700
C                                                                       CHE01710
CC                  EVALUATE THE N-TH LINEAR CONSTRAINT                CHE01720
C                                                                       CHE01730
          DO  2035 I=1,IXMAX                                            CHE01740
          DEL = DBLE( GLCNST(N,I) )                                     CHE01750
          DBI = DBLE( X(I) )                                            CHE01760
C                                                                       CHE01770
CC                  CHECK FOR NEAR 0 VALUES TO PREVENT UNDERFLOW.      CHE01780
C                                                                       CHE01790
          IF( DABS(DBI) .LT. 1.0D-39 ) GO TO 2035                       CHE01800
          IF( DABS(DEL) .LT. 1.0D-39 ) GO TO 2035                       CHE01810
          DSUM = DSUM + DEL * DBI                                       CHE01820
 2035     CONTINUE                                                      CHE01830
          XHI = BL(N)                                                   CHE01840
          XX = SNGL(DSUM)                                               CHE01850
C                                                                       CHE01860
CC                  IS X INFEASIBLE WITH RESPECT TO CONSTRAINT N        CHE01870
C                                                                       CHE01880
          IF( XX .GT. XHI ) GO TO 2094                                  CHE01890
          WRITE(NWRITE,2993) N, XX                                      CHE01900
 2993 FORMAT('0 THE VALUE OF  LINEAR CONSTRAINT NUMBER',I3,', IS:',1PE15CHE01910
```

```
     1.6)                                                                    CHE01920
           GO TO 2040                                                        CHE01930
 2094      WRITE(NWRITE,2994) N,XX                                           CHE01940
 2994      FORMAT('0 * * * * * * X IS NOT FEASIBLE WITH RESPECT TO LINEAR CCHE01950
     1ONSTRAINT NUMBER',I3,' VALUE OF CONSTRAINT IS:',1PE20.7)              CHE01960
 2040 CONTINUE                                                             CHE01970
C                                                                          CHE01980
2042  WRITE(NWRITE,2942)                                                   CHE01990
 2942 FORMAT('0 THE BOUNDS, AND ALL THE NON-LINEAR AND LINEAR CONSTRAINTCHE02000
     1S HAVE BEEN CHECKED.',//,'  NEXT THE VALUES OF THE PARTIAL DERIVATCHE02010
     2IVES OF THE OBJECTIVE FUNCTION AND THE CONSTRAINTS ARE LISTED.')     CHE02020
C                                                                          CHE02030
      DO 2050 N=1,IXMAX                                                    CHE02040
         XX = DELFX( N, X )                                                CHE02050
         WRITE(NWRITE,2950) N, XX                                          CHE02060
 2950 FORMAT('0 THE VALUE OF DELFX(',I2,', X) IS:',1PE15.6)                CHE02070
 2050 CONTINUE                                                             CHE02080
      IF( IGCNT .EQ. 0 ) GO TO 3000                                        CHE02090
      DO 2060 N1=1,IGCNT                                                   CHE02100
         DO 2060 N=1,IXMAX                                                 CHE02110
              XX = DELGIX( N, N1, X)                                       CHE02120
              WRITE(NWRITE,2960) N, N1, XX                                 CHE02130
 2060 CONTINUE                                                             CHE02140
 2960 FORMAT('0 THE VALUE OF DELGIX(',I2,",',I2,', X) IS:',1PE15.6)        CHE02150
 3000 STOP                                                                 CHE02160
      END                                                                  CHE02170
```

LIST OF REFERENCES

1. Zoutendijk, G., Methods of Feasible Directions, Elsevier, 1960.

2. Arden, B. W. and Astill, K. N., Numerical Algorithms: Origins and Applications, Addison-Wesley, 1970.

3. Hadley, G., Linear Programming, Addison-Wesley, 1962.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3.	Department Chairman, Code 55 Department of Operations Research and Administrative Sciences Naval Postgraduate School Monterey, California 93940	2
4.	Assoc Professor G. T. Howard, Code 55Hk Department of Operations Research and Administrative Sciences Naval Postgraduate School Monterey, California 93940	10
5.	Assoc Professor W. M. Raike, Code 55Rj Department of Operations Research and Administrative Sciences Naval Postgraduate School Monterey, California 93940	1
6.	Professor D. G. Williams, Code 0211 Director, W. R. Church Computer Center Naval Postgraduate School Monterey, California 93940	3
7.	LCDR J. Douglas Harrison Commander THIRD Fleet Fleet Post Office San Francisco, California 96610	1
8.	Chief of Naval Personnel Pers 11b Department of the Navy Washington, D. C. 20370	1

Thesis
H29329
c.1

155153

Harrison
A computer code for solving medium sized non-linear programming problems by the method of feasible directions.